Technische Universität München
TUM School of Engineering and Design

Digital Twin Modeling of Existing Bridges Using Optimization Algorithms and Artificial Intelligence Techniques

Mohammad Saeed Mafipour

Vollständiger Abdruck der von der TUM School of Engineering and Design der Technischen Universität München zur Erlangung eines

Doktors der Ingenieurwissenschaften
(Dr.-Ing.)

genehmigten Dissertation.

Vorsitz:
Prof. Dr.-Ing. Christoph Holst

Prüfende der Dissertation:
1. Prof. Dr.-Ing. André Borrmann
2. Prof. Dr.-Ing. habil. Xiaoxiang Zhu

Die Dissertation wurde am 22.03.2024 bei der Technischen Universität München eingereicht und durch die TUM School of Engineering and Design am 28.06.2024 angenommen.

Abstract

In the transportation system of developed countries, there is a large stock of bridges requiring substantial attention for long-term operation. Digital Twins (DTs) provide a promising solution for the maintenance and operation of bridges, thanks to their ability to mirror physical/structural conditions. A bridge DT generally consists of a geometric-semantic model whose creation, however, requires extensive manual effort. Photogrammetry and Laser Scanning are two primary geodetic techniques that can capture bridges' geometric status, leading to Point Cloud Data (PCD). Point clouds are sparse representations of real-world objects and do not inherently convey surface characteristics or geometry. To recognize the underlying geometry and provide a volumetric representation, they need to be abstracted and reconstructed properly. In current practice, the geometric digital twinning of bridges from their point clouds is conducted manually, thus increasing the costs associated with model creation. To benefit from the advantages of bridge DTs, the costs associated with the digital twinning of bridges must be decreased. This research aims to automate the geometric digital twinning process of bridges from their PCD using Artificial Intelligence (AI) techniques and optimization algorithms. The process of geometric digital twinning consists of two major steps known as *semantic segmentation* and *model reconstruction*. This thesis proposes two modules automating each part. The first module addresses the automated semantic segmentation of bridge PCD by proposing a novel Deep Learning (DL) architecture, coined a Multiscale Spatial Feature Descriptor Network (MSFD-Net). This model encodes the global, local, and relative features of points and decodes these features in various scales to determine the class label of points. MSFD-Net is tested using the PCD of ten bridges located in Bavaria, Germany, to segment the bridges' entire point cloud into partial point clouds belonging to four classes, namely *Background, Railing, Deck,* and *Abutment.* The results of applying MSFD-Net demonstrate an Overall Accuracy (OA) of 96.97 % and a mean Intersection over Union (mIoU) of 91.57 %. To automate the second part, parametric modeling is selected as a solid modeling approach to create the 3D geometric model of bridges from their segmented PCD. Following the concept of reverse engineering with parametric modeling, Parametric Prototype Models (PPMs) are proposed as tools to extract parameter values from point clouds. A local and global optimization problem is defined to adjust and assemble PPMs into an integrated model. These optimization problems are further augmented by new definitions to address point clouds with a large amount of occlusion. The proposed approach is validated by applying it to the segmented point cloud of ten single-span RC bridges as well as more complicated geometries that exist in multi-span bridges. The results show that the proposed approach can generate the parametric model of bridges with a Mean Absolute Error (MAE) of 8.43 cm. The prospective benefit of this approach for end-users is the massive reduction in modeling time and prevention of unwanted errors that commonly occur in practice.

Acknowledgements

"Once, I asked God to fill my heart with faith, and He did!"

I would like to express my sincere gratitude to the many individuals and institutions that have supported me throughout my Ph.D. Completing this thesis would not have been possible without their unwavering support and encouragement.

I deeply appreciate my supervisor, Professor André Borrmann, for his guidance, patience, and invaluable insights. Throughout the research project, I greatly benefited from his guidance, especially when he generously shared his exceptional scientific expertise and contributed to enhancing the research quality. I am also grateful for the financial support provided by the Federal Ministry for Digital and Transport - BMDV, which enabled me to pursue this research.

I wish to acknowledge the efforts of my fellow Ph.D. candidates and research colleagues at the Technical University of Munich, whose discussions and constructive comments made the research journey more fulfilling. I would also like to thank my friends for their support, understanding, and the occasional humor that provided much-needed relief during the challenging times.

My earnest thanks go to my family for their unwavering faith in my abilities and constant encouragement. Their love and support have been my pillars of strength.

Lastly, I extend my appreciation to the staff and resources at the Technical University of Munich for facilitating a conducive research environment.

This thesis is a testament to the collaborative spirit and shared wisdom of all those mentioned above. While my name is on the cover, this work is a collective effort, and I am deeply grateful to everyone who has played a part in its completion.

M. Saeed Mafipour
July, 2024

Contents

[1]Significant parts of this chapter have been previously published in the Journal of Automation in Construction. The paper can be accessed at `https://www.sciencedirect.com/science/article/pii/S0926580523003618`.

Acronyms

AASHTO	Association of State Highway and Transportation Officials.
AEC	Architecture, Engineering, and Construction.
AECOM	Architecture, Engineering, Construction, Operations, and Management.
AI	Artificial Intelligence.
ASCE	Americal Society of Civil Engineers.
BIM	Building Information Modeling.
BMS	Bridge Management Systems.
BrIM	Bridge Information Modeling.
CAD	Computer Aided Design.
CNN	Convolutional Neural Network.
CSG	Constructive Solid Geometry.
DL	Deep Learning.
DM	Digital Model.
DS	Digital Shadow.
DT	Digital Twin.
FM	Facility Management.
FPS	Furthest Point Sampling.
IFC	Industry Foundation Classes.
IoT	Internet of Things.
LiDAR	Light Detection and Ranging.

LoD	Level of Detail.
ML	Machine Learning.
MLS	Mobile Laser Scanning.
MSFD-Net	Multiscale Spatial Feature Descriptor.
MVS	MultiView Stereo.
NASA	National Aeronautics and Space Administration.
NBIS	National Bridge Inspection Standards.
NLP	Natural Language Processing.
NN	Neural Network.
O&M	Operation and Maintenace.
PCA	Principal Component Analysis.
PCD	Point Cloud Data.
PPM	Parametric Prototype Model.
RC	Reinforced Concrete.
RS	Random Sampling.
SfM	Structure from Motion.
TLS	Terrestrial Laser Scanning.
UGS	Uniform Grid Sampling.
UK	United Kingdom.
US	United States.

1 Introduction

This thesis aims to automate the geometric digital twinning process of *existing* bridges from Point Cloud Data (PCD) by Artificial Intelligence (AI) techniques and optimization algorithms. To this end, the author initially breaks down the title into its constituent elements, offering brief definitions for each component. These components are further covered in this chapter more comprehensively. This approach ensures that the reader gains a clear and accurate comprehension of the primary objectives of the research.

Digital twin modeling of...

The author defines a Digital Twin (DT) as a digital replica capable of simulating and representing a physical object within a digital environment. A DT must stay connected to the physical object to handle bidirectional updates; otherwise, this concept downgrades to a digital model at a lower integration level. The frequency of these updates, however, can be different depending on the requirements of the physical object and DT. This thesis only addresses the geometric modeling process of DTs, not other sections, such as modeling metadata and attachment of other semantics, such as reinforcement details or flawed/defective areas. Thus, the main objective of this research is to generate a Digital Twin (DT) that stays geometrically connected to the physical object and can communicate properly.

...existing bridges...

In the scope of this thesis, bridges are considered the object/asset of interest for digital twinning. The author addresses the core concepts and definitions of DTs in the following sections and sheds light on the necessity of using DTs in the Architecture, Engineering, Construction, Operations, and Management (AECOM) industry. Furthermore, the author describes bridge DTs and elaborates on their potential advantages, especially in the Operation and Maintenace (O&M) phase of bridges. This research focuses on the existing bridges that are already available in the transportation network of countries, more specifically, single-span Reinforced Concrete (RC) bridges, as they constitute a large number of existing bridges across developed countries. Furthermore, the current geometric status of the bridge (post-construction) is targeted, i.e., the as-is or as-built status, not as-designed. Data acquisition techniques such as laser scanning and photogrammetry can be employed to capture the current geometric status of existing bridges. Both methods lead to the bridge Point Cloud Data (PCD) that will be the main data source used in this research to create the bridge DT. Thus, this thesis aims to automate the geometric digital twinning of existing bridges from their PCD.

...using optimization algorithms and artificial intelligence techniques.

PCD of bridges must be processed and enriched further for generating an efficient geometric DT containing geometric details such as dimensions and volumes. The terms *optimization algorithms* and *AI techniques* forming part of the title are the methods employed to automate the geometric modeling process of bridges from PCD. Optimization algorithms are practical in geometric modeling from PCD, systematically organizing and enhancing the data to accurately represent the bridge's geometry. These algorithms streamline the extraction of relevant geometric features, such as the dimension of the bridge components and their spatial dependency. Simultaneously, integrating AI techniques brings a higher level of automation to the geometric modeling process. Machine learning (ML) algorithms can detect patterns within the PCD, learning from diverse datasets to recognize and categorize various bridge elements. This ability to automatically identify and interpret structural components facilitates the creation of the bridge DT model.

This chapter briefly introduces the requirements, challenges, and hypotheses in creating a bridge DT and proposes a method to automate the digital twinning process of existing bridges. This concise thesis representation aims to familiarize readers with the problem at hand and its proposed solution. The following chapters will further elaborate on these concepts, ensuring a thorough comprehension encompassing all relevant details.

1.1 Requirements

The fundamental premise of the thesis is that a geometric-semantic model forms the core of a bridge DT. This model is presented in a 3D digital environment where the volumetric body of bridge elements can be observed, measured, and enriched with metadata collected from the structure. It can represent the current status of the bridge and the spatial relationship between bridge elements. It can also be connected to Internet of Things (IoT) devices, be enriched with semantic information, and communicate with the existing Bridge Management Systems (BMS). These features enable bridge DTs to act as an efficient platform, especially in the O&M phase of bridges where the conventional methods cannot solely cover the large number of existing bridges.

In the development of the geometric-semantic model for the bridge DT, it is required to ensure an accurate representation of geometric parameters. A mere geometric demonstration, without precisely defined parameters, may compromise the practical efficiency of the DT. An ideal geometric bridge DT must be represented with a finite number of parameters controlling the bridge dimensions. This representation links the bridge DT to the actual bridge for accepting geometric updates and reporting geometric measurements, i.e. geometric imports and exports. This abstract representation also ensures that the model has been designed logically following the bridge engineering requirements governing the bridge design. Thus, injecting/integrating the bridge engineering concepts in generating the bridge geometric model can be expressed as one of the primary requirements in creating a practical bridge DT.

In addition to accurately defining parameters, it is crucial to determine their values precisely. A bridge DT might be similar in shape or type to the physical bridge; however, it may still require thorough attention to ensure that the assigned parameter values are precise. Any inaccuracies in determining these values can impede the practical application of the DT, hindering its effectiveness in mirroring the actual bridge and potentially leading to inaccurate assessments and decision-making. Furthermore, the parameters and values obtained from each bridge element must be aligned with their neighboring elements to generate the integrated/assembled 3D model required for a bridge DT. Otherwise, the precise piece-wise definition of each bridge element in terms of parameters and values might not be accurate for the entire bridge.

1.2 Challenges

The maintenance process of bridges can be supported with DTs; however, the manual creation of the required model for digital twinning is not only costly but also error-prone. Currently, the costs associated with creating bridge DTs outweigh the short-run benefits of the model. Hence, transportation authorities rely primarily on conventional management systems, which might lack the comprehensive insights necessary for efficient O&M of bridges. To bridge this gap, there's a critical need for advancements in automation technologies that streamline the generation of digital twin models. By leveraging cutting-edge techniques such as Machine Learning (ML), Artificial Intelligence (AI), and advanced data processing algorithms, the process of creating and updating DTs can become more cost-effective, accurate, and scalable. Automation promises to reduce the manual effort involved in model creation, thereby mitigating the financial burden and improving the accessibility of DT technology for bridge management. Cost-effective and automated methods for digital twinning provide invaluable insights, fostering proactive maintenance strategies and ensuring the long-term sustainability and safety of bridges in civil infrastructure networks.

To benefit from the potential advantages of bridge DTs, the cost associated with modeling must be decreased to a large extent. Semantic segmentation and model reconstruction are two essential yet costly steps in the geometric digital twinning of bridges. Following the requirements and specifications of a geometric DT, it needs to be capable of receiving and handling geometric updates to stay geometrically connected to the physical asset. Among the existing solid modeling approaches, parametric modeling provides an access point through which geometric updates can be handled efficiently. Despite the advantages of parametric modeling in dynamic shape updating, this approach is expensive and also requires much manual effort to define geometric constraints and parametric dependencies among components. This further intensifies the modeling expenses associated with bridge DTs.

Considering the aforementioned cases, the geometric digital twinning problem of bridges can be broken down into two subproblems, including *semantic segmentation* and *parametric modeling*. Semantic segmentation is the initial step in geometric digital twinning, where the input point cloud is divided into the point cloud of bridge elements. This step distinguishes the points belonging to the bridge structure and determines the type of bridge elements. Through semantic segmentation, the initial problem is simplified from the entire bridge point cloud to the point cloud of bridge components.

Parametric modeling is the next step, addressing the modeling process of bridge components from their segmented point clouds. The resulting model from this approach must be parametric, meaning that the volumetric models should be capable of updating their shapes depending on the input value of parameters. They must also contain geometric constraints to control the object geometry while being updated. These models should also incorporate the parametric dependencies among the bridge components to generate a consistent and coherent bridge model. Most importantly, the bridge components must be described with a finite number of parameters, each controlling a dimension.

The high complexity arising from these models limits the scalability and practical application in digital environments. This complexity can limit the seamless integration of digital twinning processes where adaptable models are necessary. Aside from the requirements addressing an ideal bridge DT, the challenges concerning the input dataset also need to be considered. PCD is an unstructured dataset without an underlying grid as in images. It often exhibits occlusion and clutter due to various factors. Limitations in sensor capabilities, like a restricted field of view or resolution, lead to incomplete data capture when objects block the sensor's view. Furthermore, areas with low light conditions, such as beneath the bridge deck, pose challenges for photogrammetry, resulting in substantial occlusion. Additionally, complex environments, such as dense vegetation around bridges, can result in clutter within point clouds.

1.3 Hypotheses

The thesis considers the following key hypotheses:

> *Hypothesis 1:* The O&M of bridges can be more effectively supported by DTs than by conventional management approaches.
>
> *Hypothesis 2:* Point clouds captured through laser-scanning or photogrammetry are a suitable basis for creating semantically rich DTs of existing bridges.
>
> *Hypothesis 3:* Deep neural networks allow the robust segmentation of a bridge point cloud into subsets representing individual bridge components.
>
> *Hypothesis 4:* A significant number of existing bridges fall into similar classes and can be represented by highly parametrized bridge models.
>
> *Hypothesis 5:* With the help of metaheuristic optimization approaches, pre-defined parametric model components can be fit into the respective point cloud segments.
>
> *Hypothesis 6:* Using highly parametrized overall bridge models ensures the geometrically and semantically coherent creation of the DT models of existing bridges.

These hypotheses contribute to a research methodology, guiding meaningful insights into the geometric digital twinning of bridges.

1.4 Proposed Solution

This thesis aims to address the aforementioned challenges by proposing an automated method for geometrically digital twinning of bridges. Figure 1.1 illustrates the research method designed to transform raw bridge PCD into the parametric model of the entire bridge structure. It consists of two major and a minor/auxiliary module emphasizing semantic segmentation of bridge point clouds and subsequent parametric modeling using the segmented point clouds resulting from the previous module.

The first module focuses on the semantic segmentation task. Considering the robust performance of Deep Learning (DL) models, their adaptability/flexibility, and less dependency on tuning problem-specific thresholds (after training), a Multiscale Spatial Feature Descriptor (MSFD-Net) is proposed that can semantically segment the input raw bridge point clouds into the point cloud of bridge elements. MSFD-Net is designed to capture three sets of features, including local, global, and relative features, and employ them in classifying points. Global features of points are defined individually in the 3D space. They are features such as (x, y, z) coordinates of points and RGB color codes that are defined for each point separately. The local features, however, are calculated considering the local neighborhood of points. These features generally provide information about the underlying surface that points represent in their local neighborhoods. Relative features also describe the pair-wise dependencies among points to represent the spatial relationships in a scene. They compensate for the lack of communication between points that are not close enough to be described in a local neighborhood. MSFD-Net decodes these features on various scales to benefit from low-level (rough) features generated in the initial blocks, as well as the high-level (fine) features extracted from the last encoding layers when assigning a class label to each point.

A minor/auxiliary module is defined between the first and second major modules to bridge the gap between semantic segmentation and parametric modeling. This module contains functions for clustering, de-noising, boundary points detection, plane detection, geometry generation, etc., and a research method to connect the first module to the second one. This minor module is responsible for processing the segmented point clouds by the first module and preparing them for the second module.

The second major module in the proposed framework addresses the parametric modeling problem. This module uses a reverse engineering approach to generate the bridge's geometric DT. Considering the desired model of the bridge, Parametric Prototype Models (PPMs) of the bridge elements are created. PPMs are dummy parametric models that can change their geometry based on the value of parameters. These models are

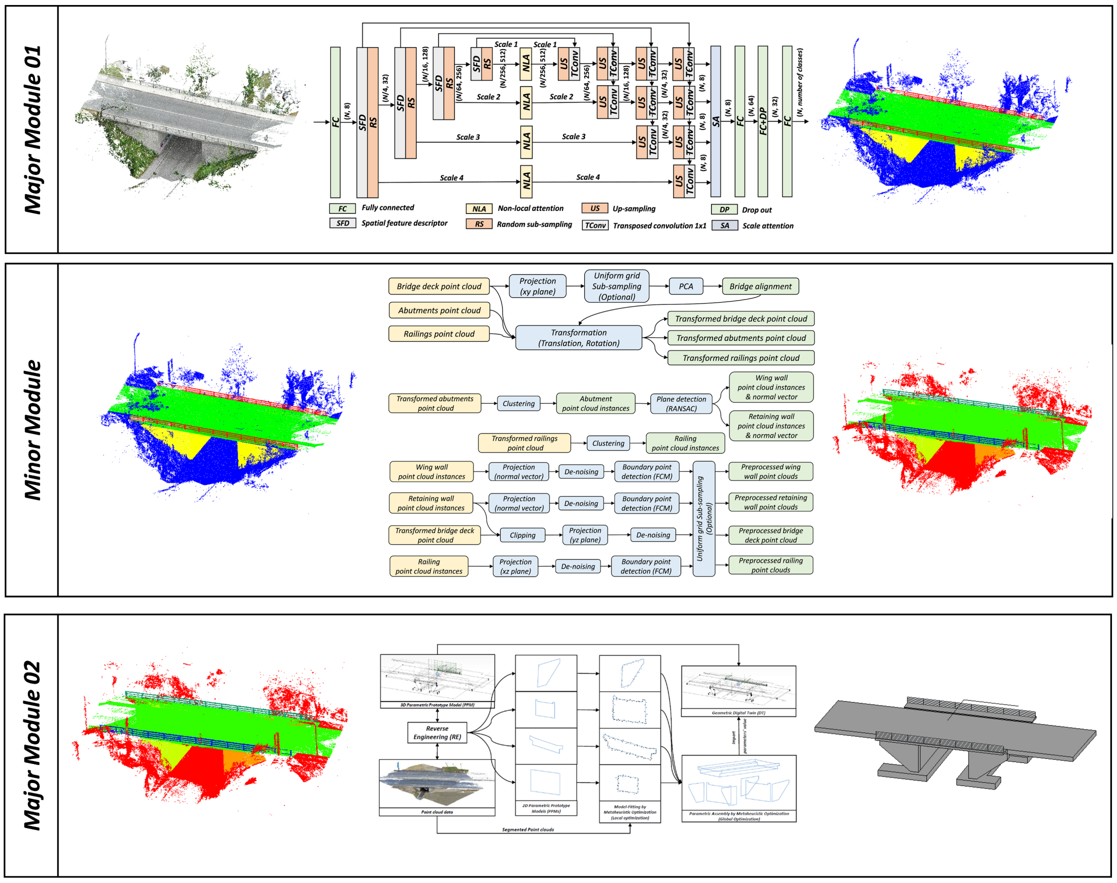

Major Module 01
FC Fully connected
SFD Spatial feature descriptor
NLA Non-local attention
RS Random sub-sampling
US Up-sampling
TConv Transposed convolution 1x1
DP Drop out
SA Scale attention
Minor Module
Bridge deck point cloud
Abutments point cloud
Railings point cloud
Projection (xy plane)
Uniform grid Sub-sampling (Optional)
PCA
Bridge alignment
Transformation (Translation, Rotation)
Transformed bridge deck point cloud
Transformed abutments point cloud
Transformed railings point cloud
Clustering
Abutment point cloud instances
Plane detection (RANSAC)
Wing wall point cloud instances & normal vector
Retaining wall point cloud instances & normal vector
Railing point cloud instances
De-noising
Boundary point detection (FCM)
Clipping
Preprocessed wing wall point clouds
Preprocessed retaining wall point clouds
Preprocessed bridge deck point cloud
Preprocessed railing point clouds
Major Module 02
3D Parametric Prototype Model (PPM)
Reverse Engineering (RE)
Point cloud data
2D Parametric Prototype Models (PPMs)
Model Fitting by Metaheuristic Optimization (Local optimization)
Geometric Digital Twin (DT)
Parametric Assembly by Metaheuristic Optimization (Global Optimization)
Segmented Point clouds

Figure 1.1: Proposed framework for geometric digital twinning of bridges.

defined considering a set of parameter values and geometric constraints. PPMs can be set up with random values in ranges obtained by bridge engineering knowledge. In this research, they are used to derive the value of parameters from the point cloud of bridge elements. For this purpose, a local optimization problem is defined to fit the PPMs into their corresponding segmented point cloud. In doing so, the randomly initialized parameter values will be adjusted to values representing the bridge element point cloud. The objective function of the piece-wise optimization problems is also augmented with new terms to address model fitting in scenarios with a large amount of occlusion. The piece-wise model fitting of bridge elements leads to the 3D model of each element. To assemble the bridge elements and generate the 3D geometric model of the entire bridge, a global optimization problem is defined to adjust the shared parameters among elements. Finally, all the parameter values after assembly are injected into the entire bridge's 3D parametric model or PPM, leading to the 3D model representing the entire bridge point cloud. This research covers all these modules to provide an end-to-end method for processing raw bridge point clouds and creating their geometric DT models.

The key contributions of this thesis are as follows:

Contribution 1: Definition of an end-to-end framework to receive raw bridge point clouds and generate their geometric DT automatically.

Contribution 2: Proposal of a novel DL model to automate the semantic segmentation process of bridge point clouds.

Contribution 3: Description of a parametric modeling paradigm to generate the parametric model of bridge components from their segmented point clouds.

Contribution 4: Proposal of a parametric assembly method to integrate the modeled bridge elements into a single bridge model with the capability of updating shape.

1.5 Thesis Structure

The thesis is structured as follows:

- *Chapter 2* elaborates on the concepts required for creating a bridge DT. It starts with defining this term and continues with its applications in the AECOM industry. It further shows how a bridge DT can facilitate the O&M process of existing bridges. Then, the required techniques for digital twinning are introduced. Finally, the current practice in the manual creation of DTs is reviewed.

- *Chapter 3* focuses on semantic segmentation and model reconstruction as two fundamental steps in creating bridge DTs and reviews the various existing approaches. It also shows how the required inputs for modeling a bridge can be estimated. It finally points out the existing research gaps that can be addressed in the thesis.

- *Chapter 4* introduces MSFD-Net (Module 01) as a DL model for the semantic segmentation of large-scale bridge point clouds. This chapter elaborates on the model architecture, its various blocks, and the concept behind them. It further shows how the extracted feature maps can be fused in various scales to generate the final feature map required for classifying points.

- *Chapter 5* elaborates on the Minor Module required for bridging the gap between semantic segmentation and parametric modeling modules. It contains the developed algorithms and methods for preparing segmented point clouds for the second major module. Clustering, de-noising, boundary points detection, and model generation are all topics and problems that are covered in this chapter.

- *Chapter 6* covers the second major module (Module 02) concerning parametric modeling of bridge elements from segmented point clouds, explaining the concepts and implementation details. It proposes PPMs as a tool to extract parameter values from bridge point clouds. It also describes the objective functions that can be written over PPMs for model-to-cloud fitting. This chapter further addresses the assembly problem for geometric digital twinning of the entire bridge, explaining how the entire bridge model can be generated.
- *Chapter 7* evaluates the major modules and quantifies their performance in terms of different statistical metrics. To validate the performance of MSFD-Net, it is compared with another state-of-the-art DL model, and expensive tests are conducted on both. In each case, the performance of the models is evaluated on previously unseen bridge point cloud samples. This chapter then uses the content elaborated in Chapter 5 to prepare the segmented point clouds for the second module. It contains a research method on how the segmented point clouds can be processed further using the proposed algorithms in the previous chapters. To validate the parametric modeling module, it is tested in geometric digital twinning of ten single-span RC bridges. In each case, the difference between the point cloud and the model is measured, and the results are presented. The core algorithms of the method are also compared with other existing methods. This chapter also demonstrates the algorithms' occlusion-resistant performance and the model's editability/adaptability for accepting geometric updates.
- *Chapter 8* finally ends the thesis with a conclusion, summarizing the thesis, discussing the research development, the significant findings, including known limitations, possible generalizations, and topics for future research.

2 Research Background

This chapter provides an in-depth exploration of the aforementioned concepts, presenting detailed definitions and insights. Additionally, it delves into the current state of the DT market, specifically emphasizing its relevance within the civil infrastructure domain. The discussion elaborates on the limitations of the conventional/traditional inspection methods and Bridge Management Systems (BMS). Furthermore, it highlights the prospective benefits that DTs can offer across different stages of construction, particularly to the O&M phase. This chapter ends by showing the current practice in creating bridge DTs and highlighting the most cumbersome steps in the geometric modeling process of bridges from their PCD.

2.1 Digital Twin (DT)

A DT represents the virtual counterpart of a physical object or system, with the ability to simulate its real-world counterpart within a digital environment. This concept has garnered significant attention across various industries, offering numerous possibilities for enhancing efficiency, productivity, and decision-making. The concept of DT has its roots in the aerospace industry, where it was initially developed to model and monitor complex aircraft systems. National Aeronautics and Space Administration (NASA) played a prominent role in popularizing this idea in the 1960s as a "living model" of the Apollo mission. Following the oxygen tank explosion on Apollo 13 and the resulting damage to the main engine, NASA utilized a variety of simulators to assess the incident and expanded the physical model of the spacecraft to incorporate digital elements [1]. This pioneering DT marked the initial instance of its kind, enabling the constant intake of data to create a model of the events prior to the incident for examination and the exploration of subsequent actions.

In the technology and data representation domain, Digital Twin (DT), Digital Model (DM), and Digital Shadow (DS) [2] are interconnected concepts; however, with various levels of integration between the digital and the physical object, as shown in Figure 2.1.

The main difference between these models relies on the data flow and the connection type between the model and the physical object. The characteristics and applications of each digital representation are as follows [3, 4]:

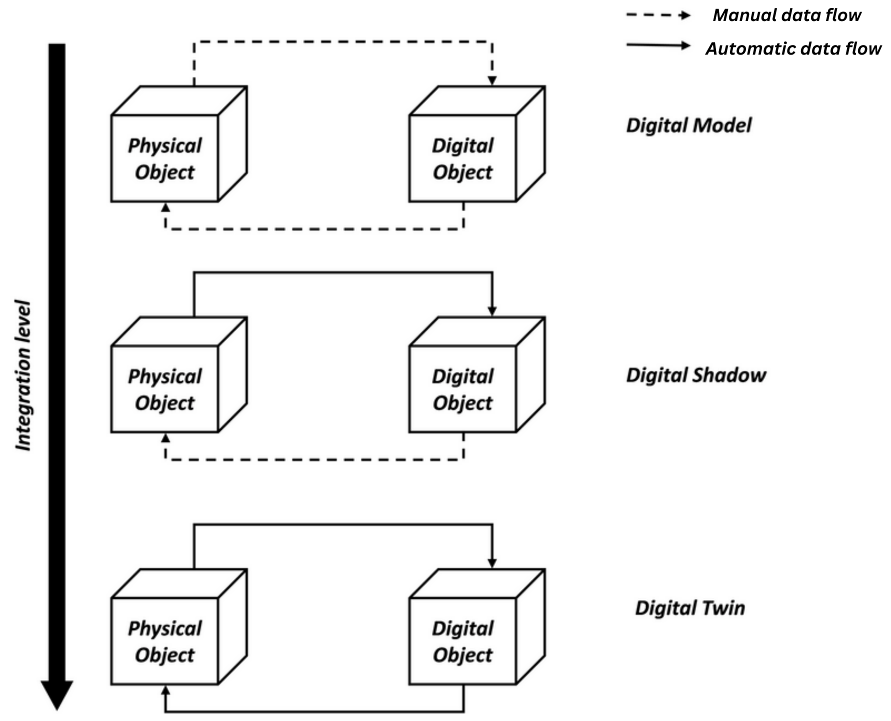


Figure 2.1: Various integration levels between a digital and a physical object [5].

- Digital Model (DM): It is a virtual representation of an object, system, or process. It is essentially a computer-generated 3D model or a mathematical representation that mimics a real-world object or system. DMs can vary in complexity, from simple 3D CAD models of a physical product to complex mathematical models used in scientific simulations. These models are primarily used for visualization, analysis, and design, and they provide a foundation for the development of more advanced concepts such as Digital Shadow (DS) and Digital Twin (DT).

- Digital Shadow (DS): It is an extension of the DM. It represents the real-time or near-real-time data and information associated with a physical object or system. Think of it as a "mirror" reflecting its physical counterpart's current state and behavior in the digital environment. These shadows are crucial in industries such as IoT and monitoring, where data from sensors, cameras, or other sources is continuously collected and used to update the model. This digital representation allows for remote monitoring, predictive analysis, and detecting anomalies in the physical world. It is used for tasks such as predictive maintenance, optimizing energy consumption, and improving overall efficiency [6, 7].

- Digital Twin (DT): It is the most advanced and comprehensive representation of a physical object or system. It combines the concept of a DM with the real-time data of a DS. A DT is a dynamic model showing the virtual counterpart of a physical object. It must be capable of communicating with the physical object to handle bidirectional updates. The interval of these updates can vary significantly and depends on the object type and its requirements. For instance, the update intervals of a jet engine can be in the range of minutes, while a city's DT might be updated only once per month or year. A DT cannot only capture the current state but also predict and simulate future behavior. This feature makes it a powerful tool for testing, analysis, and decision-making. DTs are used across various industries, from manufacturing to urban planning and space exploration [8, 9]. For instance, in manufacturing, the DT of a machine can simulate its performance, identify potential issues, and recommend optimization strategies, all prior to physical changes.

Over the years, the advancement of DT technology has been primarily led by progress in computing power, sensor technology, and connectivity. According to Grand View Research (GVR) [10], the DT market in 2023 has been estimated at $16.75 billion and is expected to increase by up to $155.84 billion in 2030 at a compound annual growth rate (CAGR) of 37.5%. The main factors impacting the huge surge of investment in DT have been claimed: 1) reducing the time and cost of development; 2) preventing unplanned downtime; 3) emerging technologies such as the IoT and Point Cloud Data (PCD); and 4) growing use of DT for maintenance purposes. DTs find utility across a spectrum of domains, including manufacturing, healthcare, smart cities, and infrastructure. The DT concept is built on several key principles that are fundamental to understanding and implementing this technology effectively. These principles include [8, 9, 11]:

- Virtual Representation: A DT is a digital or virtual replica of a physical object, system, or entity. It should accurately represent the physical counterpart in terms of its geometry, behavior, and attributes. This virtual representation forms the foundation of the DT concept.

- Real-Time Data Integration: DTs are not static models but dynamic ones that are continously updated with real-time data from sensors, IoT devices, or other sources. This real-time data integration ensures that the DT remains a current and accurate reflection of the physical system.

- Simulation and Analysis: DTs are not just passive representations but active systems capable of running simulations and analyses. These simulations can model the behavior of the physical entity, allowing for predictive analysis, what-if scenarios, and testing in a risk-free virtual environment. This capability is particularly valuable for optimization and decision-making.
- Historical Data Repository: In addition to real-time data, DTs accumulate historical data over time. This historical context provides a valuable resource for retrospective analysis, trend identification, and long-term planning. It aids in comprehending how the physical system has evolved and what changes or improvements have been made.
- Interconnectivity: DTs are often part of a broader network. This interconnectivity allows for data sharing and collaboration between various models and real-world systems. It can lead to more comprehensive insights and better coordination.
- Lifecycle Management: DTs typically cover the entire lifecycle of a system, from design and development to O&M. This ensures that DTs are valuable not only during the initial phases but throughout the system's existence, aiding in maintenance, optimization, and operation.
- Cross-Domain Integration: DTs can integrate data and models from multiple domains, allowing for a holistic view of complex systems. For example, in the context of a smart city, a DT might integrate data from transportation, energy, and environmental systems to enable comprehensive urban planning.
- Human Interaction: While DTs are driven by data and automation, they also accommodate human interaction. Engineers, operators, and decision-makers can interact with DTs through user interfaces to monitor, control, and analyze systems.
- Security and Privacy: With the influx of data and connectivity, security and privacy are crucial principles. DTs must implement robust security measures to protect data and ensure the privacy of sensitive information, especially when dealing with personal or confidential data.
- Scalability: DTs can scale from representing individual components to entire systems or even entire cities. The technology should be scalable to accommodate the complexity and size of the systems it represents.

These key principles collectively enable the creation and operation of DTs, making them a powerful tool for a wide range of industries and applications, from manufacturing to smart cities and infrastructure management.

A DT is a meticulously crafted concept defined to meet a set of requirements and specifications essential to fulfill its anticipated applications [9]. This definition serves as the blueprint for creating a virtual counterpart that accurately mirrors a physical object, system, or entity in a digital format. These requirements and specifications are the foundation upon which the DT is built, setting the boundaries and objectives that guide its development and operation. They encompass a wide range of factors, including the Level of Detail (LoD) to be replicated, data accuracy, the parameters to be monitored, and the extent of interactivity it should offer. For instance, when constructing a DT for a manufacturing plant, the requirements might specify the need for real-time monitoring of machine performance, predictive maintenance capabilities, and the ability to simulate various operational scenarios. In contrast, a DT for a smart city infrastructure might focus on traffic management, energy efficiency, and urban planning.

The precision and interconnectivity of the DT representation are intrinsically linked to these requirements, ensuring that it adequately replicates the physical entity. Thus, the more comprehensive and specific the requirements and specifications, the more effective and valuable the DT becomes in facilitating decision-making, optimizing processes, and enhancing the overall understanding of the physical counterpart.

2.2 DT in the AECOM Industry

DTs potentially have various applications in the AECOM industry. They can be used to model and manage buildings and civil infrastructure assets throughout their entire lifecycle, from Design and Construction to O&M. In the AECOM sector, the concept of "DT" has been expressed as an evolved version of Building Information Modeling (BIM) or Bridge Information Modeling (BrIM) with capabilities and differences in the linkage of the model to the real asset. As DTs have the ability to inherit all the features of the conventional models, the concepts of BIM and BrIM are described first and extended then to that of the DT.

BIM plays an increasingly prominent role in the AECOM industry by providing the geometric-semantic representation of assets. BIM is a 3D modeling and information management method that integrates various aspects of a construction project into a single and coherent model [12]. This geometric-semantic model provides a 3D representation of data, allowing stakeholders to visualize the design and realize the contextual dependencies among components/elements. This 3D model improves design comprehension and communication among the involved members and parties in the project to end up with more accurate decisions. Hence, risk mitigation can be expressed as one of the significant features of a BIM model as it provides a platform for assessing the design changes on the project. Beyond being a geometric model, BIM includes semantic information about the project, such as materials, schedules, and performance specifications. This data can also be updated and shared, ensuring all the stakeholders use an up-to-date version of the data. BIM models also serve as tools for as-built documentation capturing the current state and changes made during the construction. This feature makes them also appropriate for the construction phase, where some changes might occur in the project implementation. It provides insight into the maintenance schedules, equipment specifications, and space utilization. Sustainability is one of the benefits of BIM by providing data on material, energy performance, and lifecycle analysis. Several studies [13, 14] show the application of BIM in reducing CO_2 emission during the life cycle of buildings as well as sustainable decision-making of building structures to achieve an equilibrium between energy efficiency and performance. Serving as a database in the post-construction phase, BIM extends its realms to the Facility Management (FM) of buildings, where it potentially improves the efficiency of data exchange in the maintenance phase of the asset [15]. However, the primary concern lies in the absence of FM during the developmental stages of BIM [16]. Considering the construction phase of buildings, BIM models are broadly categorized into the following three classes [17]:

- *As-designed*: It signifies the virtual representation and documentation of a building at the point where the design has been finalized. It also deals with integrating the collected data by the design team before construction. An as-designed BIM provides detailed information about the architectural, structural, and mechanical design. These models encompass 3D visualizations, technical specifications, materials, and other relevant data. As-designed BIM models can be referred to as a reference point throughout the construction process, allowing more productive communication among architects, engineers, contractors, and clients.

- *As-built*: It is the BIM model generated after completion of the construction process, reflecting the accurate and up-to-date status of the physical structure, systems, and components of the project. These models result in the precise documentation of the building at a specific point in time. As-built models can be achieved by either applying the changes to the as-designed model during the project's construction phase or surveying the implemented building and creating a BIM model right after the construction phase. The up-to-date representation of these models aids them in finding many applications in FM, including rehabilitation, operation, and maintenance. Despite the various applications of as-built models, they are less common than as-designed models in current practice. This is mainly due to the challenges and costs associated with the creation and semantic enrichment of the models.

- *As-is or as-maintained*: The real-world assets might undergo some changes during their service life (e.x., two rooms in a building might be merged). As-is BIM refers to the model representing the current status of the building at a point in time after construction. Creation of an as-is model necessarily requires on-site data collection using techniques such as laser scanning or photogrammetry. Compared with as-built models, they are less common and more costly; however, they provide a highly precise map for well-informed decision-making. As-is models are also highly practical for condition assessment and analysis of the structure after aging and experiencing changes. The manual creation of these models and semantic enrichment is labor-intensive and error-prone. They, thus, are the least common category of BIM models that can be found currently.

Similar to BIM for buildings, BrIM addresses the digital representation and management of bridge infrastructure in the entire life cycle of the structure. In the civil infrastructure domain, bridges have been widely investigated for developing BrIM in the as-designed, as-built, and as-is phases. A detailed comparison by Kumar et al. [18] illustrated the significant advantage of using BrIM over conventional approaches by implementing three bridge projects by spending five times less time. In addition to the as-designed and as-built phases of bridges, BrIM has been highly beneficial in the as-is phase for data acquisition and structural health monitoring (SHM) [19, 20]. BrIM can facilitate the inspection and evaluation process of bridges [19], as it is highly effective in the 3D representation and documentation of flawed and defective areas of bridges. BrIM can be also used in connection with BMS for documentation, structural analysis, and condition assessment [19]. The same applies to manual inspections and the localization of identified defects and damages. Compared with traditional 2D drawings, BrIM provides a more comprehensive representation in a 3D environment with the capability of continuous semantic enrichment at various levels. This model can be shared with the involved teams in the project and is used for more accurate decision-making on the possible rehabilitation of the structure. Applications of BrIM can be summarized as follows [21, 22, 23, 24, 25]:

- Inspection: BrIM provides a geometric-semantic model that can be used to plan, gather, archive, and apply damage-related data based on the location of a surveyed bridge. In this context, Unmanned Aerial Vehicles (UAVs) for structural monitoring and maintenance, Remotely Piloted Aircraft Systems (RPAS) to enhance operational efficiency and Light Detection and Ranging (LiDAR) systems for automated crack detection can be employed, and their results are linked to the model as shown in Figure 2.2.

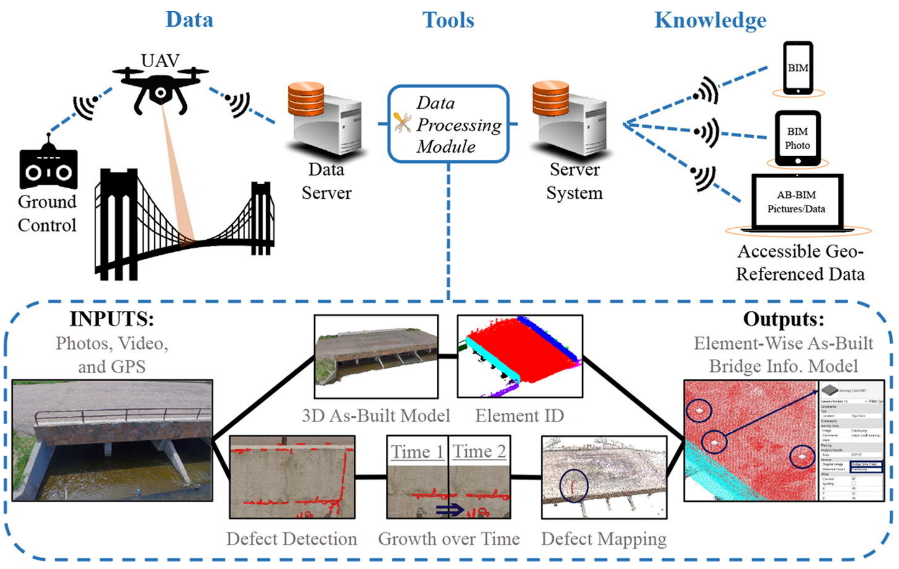


Figure 2.2: Bridge inspection using UAVs [26].

- Accelerated bridge construction: BrIM can be applied in the entire spectrum of construction activities within the project, starting from conceptual design to the construction phase. The utilization of BrIM improves the efficiency and sustainability of bridges by optimizing material consumption and decreasing the need for information and change requests. BrIM can be employed for documentation and generating related diagrams, conducting design analyses, and planning the positioning of precast concrete components. Furthermore, it can assist in the rapid implementation of bridges by determining the surrounding ground level and facilitating the transportation and placement of the superstructure components.

- Structural analysis: A BrIM model can be used as input for structural software for analysis and design of the structure (Figure 2.3). This integration allows for the creation of structural positioning diagrams within the model as well as the automated generation of reinforcement details based on the results of the structural analysis. Additionally, BrIM can be employed for Finite Element Modeling (FEM) and further analysis of the bridge.

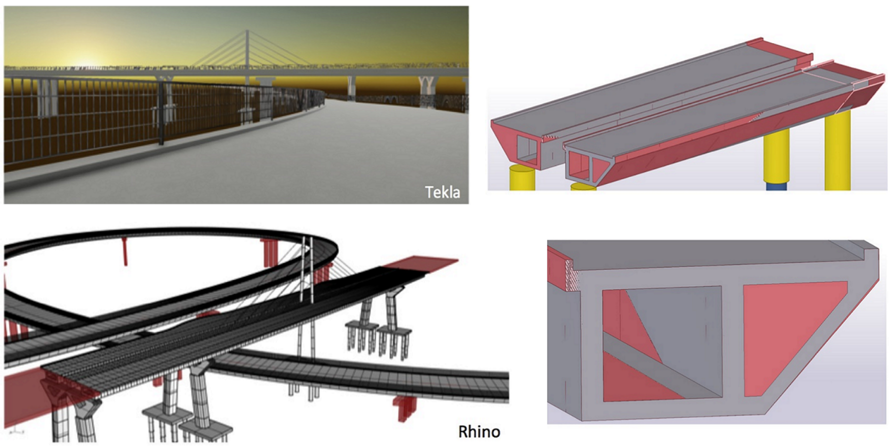


Figure 2.3: BrIM model imported in structural analysis software [27].

- Rehabilitation and repair: Numerous bridge structures require rehabilitation and repair due to changing weather conditions, the aging of the structure, and various other considerations. Aging bridges, especially those spanning rivers and tunnels, require rehabilitation actions to meet conditions and safety requirements. BrIM allows multiple experts from different domains to access and work on the target structure simultaneously. This live monitoring of construction activities and other relevant aspects ensures that up-to-date geometric-semantic information is being used in the maintenance phase of the structure.

- Management and maintenance: The resulting model of BrIM can be connected to a BMS for automated extraction and retrieval of documents as shown in Figure 2.4. This system can address bridge degradation and determine the priority for maintenance actions, identifying fragmentation damage via point cloud analysis and incorporating damage components. This integration not only improves the decision-making process but also enhances the accuracy of assessments by leveraging advanced analytical techniques provided by BrIM and the organizational capabilities of the BMS.

In the context of AECOM, the concepts of DT and BIM/BrIM have been connected and brought together [29, 30]. For instance, Figure 2.5 shows the life cycle of a DT in different stages of design, construction, and operation. Each stage has also been interconnected with a type of BIM model such that the end point of the design stage has

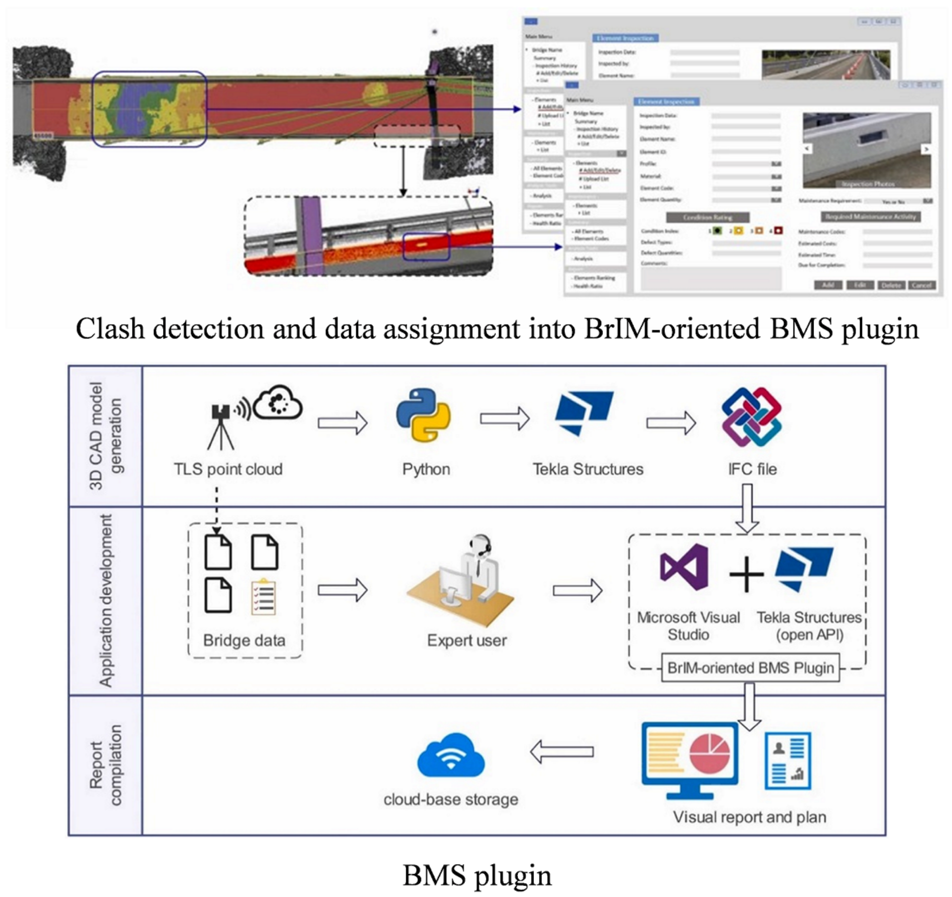


Figure 2.4: Bridge management through BrIM and laser scanning where the bridge data is attached to the model to visually represent clashes [28].

been devoted to the as-designed BIM, the endpoint of the construction stage to the as-built, and after that on to the as-is. This diagram also has a horizontal axis representing the border between the DT and the physical asset, herein called a Physical Twin (PT). Following this diagram, a DT undergoes an evolution since the start of the project as follows [29]:

- *Foetal DT*: During the design phase, the asset's design team works on the conceptual plan. The initial DT of the asset encompasses both product and process information. This file serves as a reference point for assessing construction outcomes and can provide guidance for maintenance purposes.

- *Child DT*: Moving on to the construction phase, the child DT includes off-site prefabricated assemblies and on-site constructed components. Consequently, the child

DT contains as-built product information and as-performed process information, reflecting the asset's physical status at different stages during construction. Over this stage, the process information accumulates in the child DT until construction is complete. Each change is updated in the asset's child DT to represent the as-is status, facilitating progress monitoring and quality control.

- *Adult DT*: In the operation stage, the adult DT remains unchanged due to the completion of construction. The asset's adult DT supports performance analysis, including aspects such as energy consumption and component maintenance. Collected data is added to the as-maintained model to develop the adult DT.

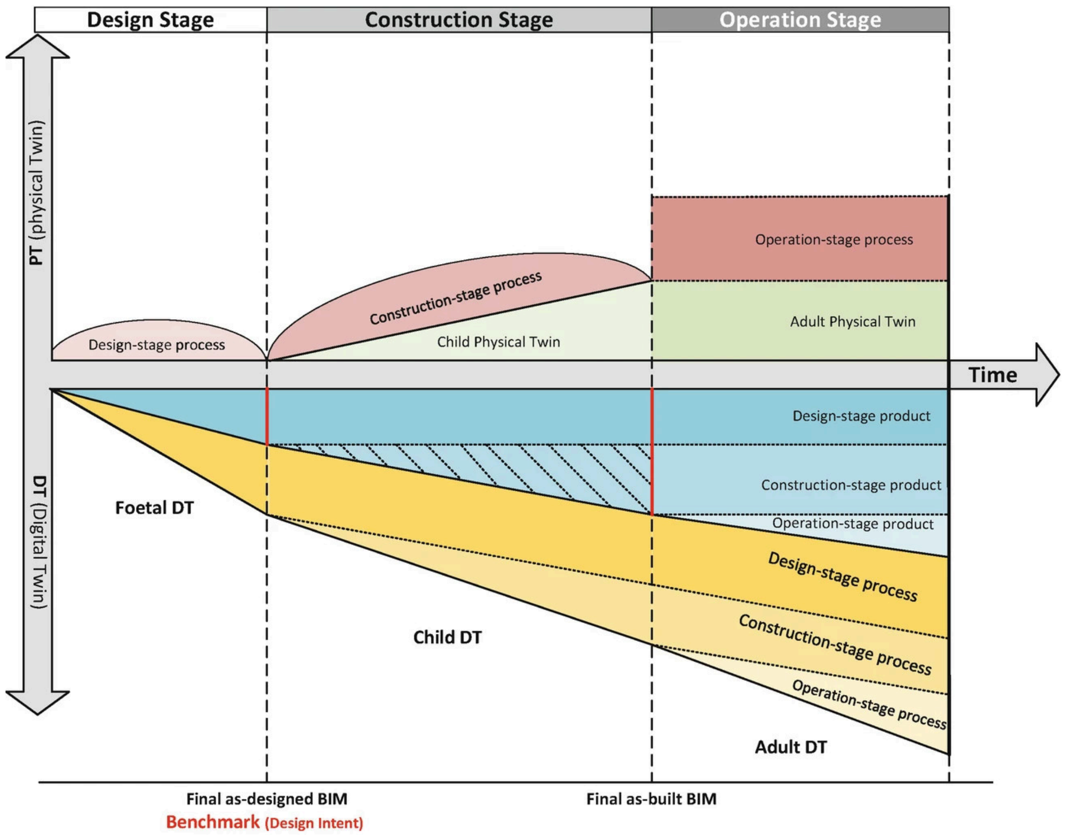


Figure 2.5: Life cycle of a DT in the design, construction, and operation stages [29].

While there are similarities and connections between a DT and a BIM/BrIM model, they have different definitions and cannot be simply mapped onto each other. Following the definition of a DT, this model requires a physical counterpart that must already exist. Nonetheless, an as-designed BIM is created before starting the construction process when no physical object is yet built. Thus, a DT cannot be defined in the pre-construction phase and is related to an as-designed BIM model. However, a DT defined at other stages, such as construction and operation, might inherit some geometric-semantic data from the as-designed phase. Considering the DT concept, there must be a bidirectional link between the physical object and the DM such that the up-to-date data can be imported and exported, and its reflection is seen on both sides. However, most of the existing as-built and as-is models still lack such a link to handle bidirectional updates.

The DT concept is highly generic as it finds its roots in the manufacturing industry [31]. This model is defined purposefully based on a set of requirements, specifications, and boundaries that might result in a model completely different than that of the as-built and as-is. For instance, the DT of a building can be a simple data model or interface for capturing the thermal changes in rooms through sensors and setting the temperature accordingly. In this example, a digital model and a physical object exist that can bidirectionally communicate through an access point, while the model is not similar to the conventional BIM models. In better words, BIM/BrIM can be considered as a specific type of DT only if they can provide an access point to handle the bidirectional data transition and execution of queries. Otherwise, the concept downgrades to a DM or DS as it loses its link for the required updates and integration.

2.3 DT in the O&M of Bridges

There is a large stock of bridges in the transportation system of developed countries requiring substantial attention for long-term operation. The United States (US) spends roughly $14.4 billion per year to address the deteriorating bridge conditions [32]. In the United Kingdom (UK), the annual maintenance cost of road networks has been estimated at around £4 billion [33]. In Germany, the maintenance cost of only federal bridges has been approximated to €350 million every year [34].

The recent Americal Society of Civil Engineers (ASCE) report card [35] shows that there is an extensive network of over 617,000 bridges in the US. Currently, 42% of these bridges are more than 50 years old, and among them, 7.5% are deficient bridges. The report card also asserts the deterioration rate of existing bridges has exceeded the rate of repair and rehabilitation as the conventional methods cannot adequately provide a mechanism for efficient coverage of all bridges. It thus recommends the annual spending on bridges to be increased from the current $14.4 billion to $22.7 billion, meaning a 58% increment; otherwise, addressing all the current deficient bridges takes until 2071. It also predicts the deterioration over the next 50 years will become increasingly overbearing, and thus, systematic bridge maintenance programs must be implemented.

Considering the prominent role bridges play in the civil infrastructure domain, National Bridge Inspection Standards (NBIS) require transportation agencies to evaluate the condition of existing bridges at frequent intervals over the service life of the structure [36]. Following the German standard DIN 1076 [37] and NBIS, different inspection stages are considered for a bridge, including routine inspections, initial inspections, inspections on special occasions, hands-on inspections, etc. A complete list of various bridge inspection types can be seen in Table 2.1.

Table 2.1: Inspection types of bridges according to the U.S. code of federal regulations [36].

Inspection	Description
Damage Inspection	An unscheduled inspection to assess structural damage resulting from environmental factors or human actions.
Fracture-Critical Member Inspection	A hands-on inspection of a fracture-critical member or member components that may include visual and other nondestructive evaluation.
Hands-On Inspection	Inspection within arms length of the component. Inspection uses visual techniques that may be supplemented by NDT.
In-Depth Inspection	A close-up inspection of one or more members above or below the water level to identify any deficiencies not readily detectable using routine inspection procedures; hands-on inspection may be necessary at some locations.
Initial Inspection	First inspection of a bridge as it becomes a part of the bridge inventory to provide all Structure Inventory and Appraisal data and other relevant data and to determine baseline structural conditions.
Routine Inspection	Regularly scheduled inspection consisting of observations and/or measurements needed to determine the physical and functional condition of the bridge, to identify any changes from initial or previously recorded conditions, and to ensure that the structure continues to satisfy present service requirements.
Special Inspection	An inspection scheduled at the discretion of the bridge owner, used to monitor a particular known or suspected deficiency.
Underwater Inspection	Inspection of the underwater portion of a bridge substructure and the surrounding.

Initial inspections are simple assessments carried out to monitor the bridge's initial condition, leading to a reference point describing the as-built conditions. Routine inspections are the main bridge assessment process. These inspections are conducted at specified intervals to ensure the bridge's structural integrity, safety, and overall condition. Routine inspections might be followed by an in-depth examination of the element in cases where the damage is not readily detectable. In critical cases, an in-depth examination of damage is required within the arms length of the bridge element. This step might need the use of monitoring systems, sensors, and data collection tools to track the bridge's performance over time continually. It might also be followed by a fracture-critical member inspection to provide further details about the flawed area.

Damage and special inspections are less frequent than routine inspections and are conducted under unique circumstances or events that might affect the bridge's integrity. Such occasions include extreme weather events, natural disasters, or incidents like vehicular collisions. These inspections ensure that the bridge's safety and structural conditions are evaluated promptly in response to specific situations. Some bridges are subjected to specific regulations or standards as well due to their unique design, location, or importance. Inspections tailored to these special regulations ensure that the bridge complies with all requirements and remains safe to use. Underwater inspections are also a special type of bridge inspection mainly conducted for bridges whose sub-structure is partially under water.

Due to the large number of bridges, the pressing necessity for various types of inspections, and the high cost of repair, corresponding agencies have developed various rating systems for prioritizing bridge rehabilitation projects [32]. In Germany, the main inspections are mostly conducted through visual assessment of the structure [38]. For every visible damage lying on the primary bridge elements such as abutments, deck, girders, and piers, four factors, including valuation criteria, stability, traffic safety, and durability, are considered, and then, a condition index in the range of 1.0 (very good condition) to 4.0 (insufficient condition) is determined, as shown in Table 2.2 [38].

Likewise, in the US and UK, operators inspect and rate a bridge following instructions and standards such as NBIS [36], Association of State Highway and Transportation Officials (AASHTO) [40], and Manual for Highway Structures [41]. The US condition rating system, however, is established within another range, spanning from 0, indicating a "Failed" state, to 9, signifying an "Excellent" condition. This system provides a structured framework for the classification of bridges, allowing for consistent assessment of the condition and performance. The ratings, as outlined in Table 2.3, aid bridge authorities and engineers in making decisions regarding maintenance, repairs, and infrastructure investment.

The inspection process of bridges conventionally leads to 2D documents containing geometric-semantic information concerning the current status of the bridge. These doc-

Table 2.2: Description of bridge condition rating in Germany [39].

Grade	Description
1.0-1.4	Very good structural condition Continue normal maintenance
1.5-1.9	Good structural condition, but may have less long-term durability Continue normal maintenance
2.0-2.4	Satisfactory structural condition, but may have less long-term durability Continue normal maintenance and consider a plan for repair
2.5-2.9	Unsatisfactory structural condition Traffic safety may be affected Structure is not sufficiently durable Continue normal maintenance and plan for repair Restrictions on traffic use or load may be needed
3.0-3.4	Critical structural condition Traffic safety is affected Structure is not durable Immediate repair is needed Restrictions on traffic use or load are needed
3.5-4.0	Inadequate structural condition Traffic safety is not adequate Structure is not durable Immediate repair or rehabilitation is needed Restrictions on traffic use or load are needed

uments are captured into a BMS. These systems are typically created internally by the managing organization of a country (with or without the assistance of private companies) or procured as off-the-shelf solutions and then customized to align with their specific requirements [43]. SIB-Bauwerke (Road Information Database – Structures) is the German BMS that receives construction data, inspection reports, and intervention history following the guideline ASB-ING (Instruction for the Road Information Database, Subsystem structural data) [44], and characteristics of damages, according to RI-EBW-PRÜF [45]. Bridge engineers and experts generally employ the collected information in the BMS to analyze and interpret data to determine a grade for the current conditions of bridges following the defined ratings in Table 2.3 or 2.2.

Despite the feasibility of utilizing this conventional approach in the O&M phase of bridges, the rising costs from visual inspection, data collection, management, and interpretation prevent transportation sectors to cover operating bridges efficiently. Furthermore, the current approach has the following limitations:

Table 2.3: NBIS ratings for bridge conditions [42].

Rating	Condition	Description
9	Excellent	New condition, no noteworthy deficiencies
8	Very good	No repair needed
7	Good	Some minor problems, minor maintenance needed
6	Satisfactory	Some minor deterioration, major maintenance needed
5	Fair	Minor section loss, cracking, spalling, or scouring for minor rehabilitation; minor rehabilitation needed
4	Poor	Advanced section loss, deterioration, spalling or scouring; major rehabilitation needed
3	Serious	Section loss, deterioration, spalling or scouring that have seriously affected the primary structural components
2	Critical	Advanced deterioration of primary structural elements for urgent rehabilitation; bridge maybe closed until corrective action is taken
1	Imminent failure	Major deterioration or loss of section; bridge may be closed to traffic, but corrective action can put it back to light service
0	Failed	Out of service and beyond corrective action

- The data collection process often results in the accumulation of information in various formats, necessitating a subsequent integration process. This integration phase involves harmonizing and consolidating data from different sources, ensuring it can be effectively and cohesively utilized for analysis and decision-making.

- Data interpretation using the current BMS, which primarily relies on 2D documents, can be laborious and complex. The reliance on 2D representations often presents significant challenges when comprehending and making sense of the data.

- The inspection process, which is mainly conducted through direct observation of the bridge components, requires a substantial investment of both time and cost. This can be even intensified in bridges that have hard-to-reach regions, such as tall piers and elevated bridge decks that are not simply accessible, as shown in Figure 2.6. These inaccessible regions compound the time and cost associated with the inspection, as specialized equipment and extensive safety measures are often necessary to carry out thorough assessments.

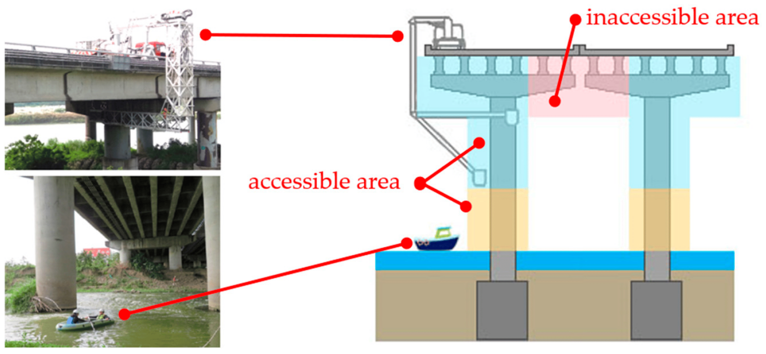


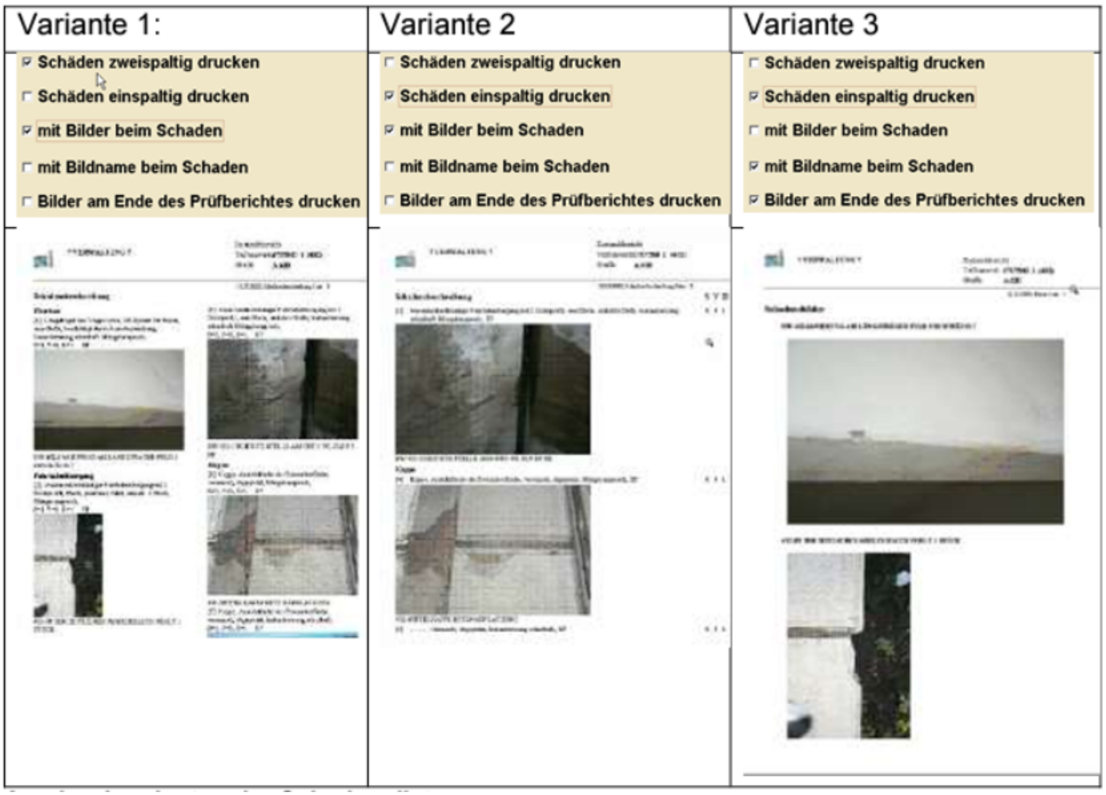
Figure 2.6: Traditional bridge inspection process [46].

- Planning bridge rehabilitation and strengthening programs only relying on the BMS leads to significant challenges. This difficulty arises from a lack of a comprehensive 3D representation of the bridge, including its intricate geometric-semantic details. Without a three-dimensional view, understanding the bridge's spatial characteristics remains limited, making it complicated to plan effective rehabilitation and strengthening strategies. Furthermore, the data collected following inspections might prove inadequate, leaving gaps in the information necessary for precise planning. This case generally results in the repetition of the inspection process. The consequences of this inefficiency are financial and impact project timelines, potentially delaying the bridge rehabilitation process.

- The complexity of the data interpretation process significantly inhibits the active involvement of experts in the decision-making process and the evaluation of the bridge's current condition. As a result, the outcomes tend to be subjective and can exhibit variations when assessed by different experts. The subjectivity arises from the inherent challenges in comprehending and interpreting data, leading to a potential lack of concurrence among experts and their varying perspectives on the bridge.

- The collected data is not available for particular bridge elements; thus, detailed information such as materials, cracks, and their locations are often captured by hand sketches. For instance, Figure 2.7 represents various variants of damage inserted into the SIB-Bauwerke BMS. Despite the 2D representation of the damage, its location, the hosting element, and the characteristics of the crack, such as its depth, length, and width, are not obvious. As a result, decision-making on the type of damage and its impact on the entire bridge structure becomes challenging.

The absence of comprehensive, detailed data may hinder the ability to make well-informed assessments and decisions in the maintenance phase of bridges.



Variante 1:	Variante 2	Variante 3
☑ Schäden zweispaltig drucken	☐ Schäden zweispaltig drucken	☐ Schäden zweispaltig drucken
☐ Schäden einspaltig drucken	☑ Schäden einspaltig drucken	☑ Schäden einspaltig drucken
☑ mit Bilder beim Schaden	☑ mit Bilder beim Schaden	☐ mit Bilder beim Schaden
☐ mit Bildname beim Schaden	☐ mit Bildname beim Schaden	☑ mit Bildname beim Schaden
☐ Bilder am Ende des Prüfberichtes drucken	☐ Bilder am Ende des Prüfberichtes drucken	☑ Bilder am Ende des Prüfberichtes drucken

Figure 2.7: Varaiants of damages in the SIB-Bauwerke BMS.

- Applied changes to the bridge following retrofitting cannot be easily documented and integrated into the conventional BMS. The challenge lies in the complexities associated with capturing and updating the BMS to accurately represent the post-retrofit state of the bridge. Changes to the bridge structure, components, or systems may involve various alterations, from structural enhancements to the installation of advanced instruments. These alterations often necessitate updates to the BMS to ensure that it accurately reflects the bridge's current state. This process can be complicated and time-consuming, requiring precise data collection, thorough documentation, and the integration of new information into the BMS. The difficulty in reflecting these post-retrofit changes in the BMS can result in a lag between the physical state of the bridge and its digital representation. This temporal disconnection may restrict efficient maintenance, management, and operation of the bridge, as decisions must be ideally based on the up-to-date conditions of the bridge. Aside from these, in many cases, a history of the applied changes is required for informed decision-making and planning further possible actions. For instance, the length of a crack, its propagation over time, and its closeness after retrofit must be available.

- Most existing BMS only provide tools for the statistical analysis of the bridge elements. In other words, there is no 3D model that can be analyzed with cutting-edge analysis software. This case is more common for old bridges without any 3D representation, while they are the major candidates for rehabilitation actions. In many cases, a bridge, after retrofitting, needs to be analyzed again to ensure the strengthening procedure has improved the performance of the bridge in sustaining the applying dead and live loads to the structure. Also, continuous monitoring of old bridges is necessary in many cases that can be achieved through IoT devices such as sensors.

To address these challenges, conventional methods for maintaining, managing, and operating bridges need to be supported with digital methods. Considering the current system's limitations, a DT can be defined for existing bridges. A bridge DT promises a substantial improvement in extending the life cycle of bridges by providing a coherent digital replica mirroring the physical reality, including the current status of the actual asset [47]. As described in Section 2.1, a DT is defined purposefully based on its anticipated applications, serving the use cases and requirements that generate a DT for a specific domain. The prominent feature of a DT is its capability to be linked with the actual asset through an access point or a communication gate to handle bidirectional updates. The interval of these updates might differ depending on the asset type and the desired use cases [47]. For bridges, adjustments to geometry may need occasional attention, especially when variations in the shape of bridge components are detected. A feasible scenario can be a bridge subjected to the dynamic load of vehicular traffic, resulting in significant deflection. In this case, addressing and managing geometric updates becomes crucial to ensure structural integrity and performance.

Nonetheless, a DT must be capable of receiving and handling the required updates to provide an up-to-date representation of the actual asset. A bridge DT can be as simple as a 2D map representing the general but up-to-date information of the bridge or as complicated as a 3D geometric model that includes all the cracks and spalling on the structure, as well as the state of the interior systems, such as pre-spanning cables. The DT can inherit all the features of BrIM, is linked with the BMS, and reflects the impact of the external factors on the structure [48]. All these features enable DT to perform as an efficient digital representation for supporting and facilitating the O&M of bridges.

A DT can be generated based on a set of requirements and specifications, and in special cases, it can converge to a BrIM/BIM model if the bidirectional link between the model and the physical object is established. Requirements are the system's goals, while specifications convey more details about the requirements. Considering the limitations of the current management, maintenance, and operation system, the absence of a 3D representation has given rise to issues concerning the comprehensive comprehension of the semantic-geometric status of bridge elements as well as their spatial relationships. It is also necessary for the structural analysis of the bridge and illustration of the flawed/defective areas on the body of the structure. Thus, 3D geometric modeling of bridges can be expressed as one of the requirements of an advanced/ideal bridge DT.

This 3D geometric representation also has some specifications. For instance, the geometric model must represent the current geometric status of the bridge and be capable of handling geometric updates. Another requirement is to connect the model to not only the actual asset (this is the core feature of a DT) but also the BMS to encompass the historical data of the bridge and handle bidirectional updates between the physical bridge and its 3D digital model. This feature solves the problems associated with the long-lasting data interpretation process and integrates various data formats through a linked 3D visualization. In this case, compatibility of the BMS with the model and secure communication can be expressed as one of the specifications.

The DT can also have the capability of connecting to IoT devices, such as sensors, to provide real-time data about the current condition of the bridge and instant communication between the physical bridge and its digital model. Such a bridge DT can provide high-level information about the dimensions of the elements and their spatial relationships, the type and characteristics of damages and their locations on the bridge, the historical data about the bridge such as the propagation of cracks, the behavior of the structure under applying loads, and a model for advanced analysis of the structure. A typical bridge DT representation can be seen in Figure 2.8, where various data sources such as 2D plans and damage pictures have been connected to the model to demonstrate their location and other semantic information on the bridge model. The linking of these resources already creates significant benefits. The legacy tabular inspection data can be enriched with spatial representations, which improves the inspection process, as the location of damages can be easily retrieved. On the other hand, the geometry model is enhanced with material and type specifications from the bridges' documentation data [48].

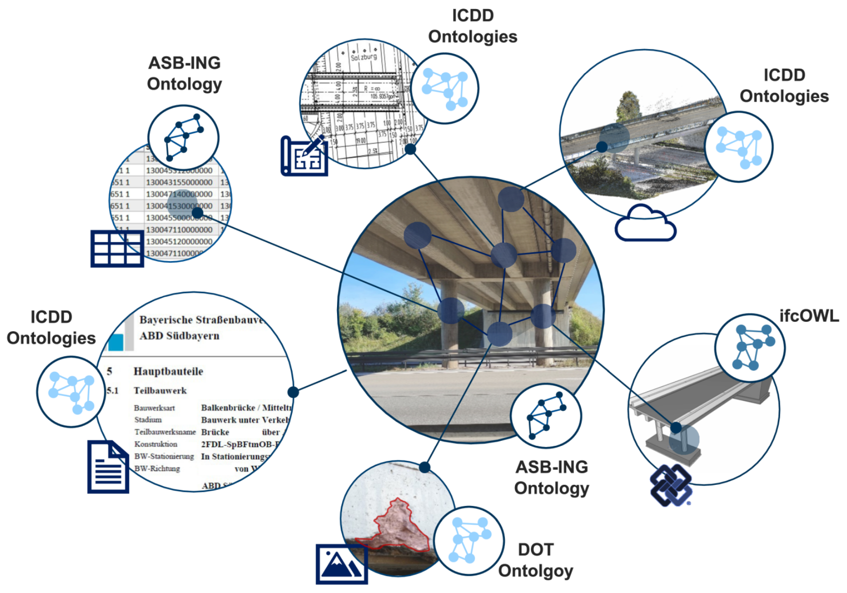


Figure 2.8: DT model linked with heterogeneous data sources [48].

2.4 Data Acquisition for Geometric Digital Twinning

To achieve a practical bridge DT that can be efficiently used in facilitating the O&M process, the bridge's 3D geometric-semantic model needs to be created. This model must also be up-to-date and represent the geometric status of the structure, including geometric constraints, dimensions, and volumes. The primary data for constructing this model consists of visual and spatial information. Although the means of collecting such data are typically limited to external surfaces, these data encompass the full potential to achieve results at a comparable level with visual observation. The outcome of the reality-capturing process is generally in 2D and 3D data formats such as images and point clouds. They can be obtained through sensors or by applying a set of transformations and processing steps to infer additional spatial details, effectively converting the 2D data into 3D data [49]. Since the selection of acquisition and transformation methods significantly impacts the quality, quantity, and characteristics of the final output, it is essential to consider a combination of both in line with the distinct requirements and specifications.

Various techniques can be employed to acquire the 2D and 3D data required for digital twinning. Photogrammetry and Laser Scanning are two primary geodetic techniques commonly used to capture existing assets due to the low manual effort required. Both techniques generate PCD, however, with varying levels of accuracy and density. In the civil infrastructure domain, these techniques have been commonly used to gather data about geometry, concrete deterioration, steel rebar corrosion, water seepage, concrete cover delamination, spalling, deflection, and cracks [49, 50]. Recently, a comparative analysis of accuracy and reliability has also demonstrated the capability of both methods in the digital twining of bridges [51].

Laser scanning, generally referred to as LiDAR, relies on the emission of laser beams toward the target object or environment. This technique employs either time-of-flight or phase-based technology to meticulously capture and record the (x, y, z) coordinates of objects, as well as the accompanying intensity data. In time-of-flight systems, the method relies on measuring the time taken for emitted light or laser pulses to bounce off objects and return to the scanner. This time measurement is then converted into distance, enabling the precise determination of the object's position in three-dimensional space. On the other hand, phase-based technology operates by examining the phase shift of emitted light or laser waves when they encounter surfaces and are reflected back to the scanner. This phase shift data is interpreted to calculate the 3D coordinates of scanned objects. Time-of-flight has been used more commonly in civil engineering applications [49].

LiDAR sensors are typically mounted on one of two types of platforms: Terrestrial Laser Scanning (TLS) systems are often mounted on tripods, while Mobile Laser Scanning (MLS) systems have sensors mounted on frames that can be carried or driven. TLS systems are commonly used for capturing extensive outdoor spaces and indoor environments, as they can measure points over distances with high precision. However, they need to be moved and set up for each measurement, resulting in separate datasets that must be registered to create a coherent point cloud. LiDAR, due to its line-of-sight limitation, may lead to significant occlusions depending on the geometry of the surroundings. MLS systems are more portable and lightweight but are less precise than TLS. Contrary to TLS, they can capture data while moving, reducing the occurrence of occlusions and increasing the coverage level.

Photogrammetry is concerned with capturing 2D images of an object from different angles and aligning the common points in the images, leading to PCD [50]. It relies on the principles of triangulation to determine the 3D location of points. It is also called Videogrammetry when capturing videos instead of photos. Photogrammetry consists of two fundamental steps: Structure from Motion (SfM) and MultiView Stereo (MVS) [52]. SfM is a method used to construct a 3D representation from a set of overlapping images captured from various angles. The SfM method was initially proposed based on computer vision techniques to estimate the camera position using multiple images [53].

The SfM process starts with the detection and extraction of distinctive features through feature matching and geometric validation. It subsequently reconstructs the object in a 3D spatial context with the intrinsic and extrinsic camera parameters for all the involved images. MVS, on the other hand, receives the outcomes of SfM and computes depth and normal information for pixels within the images, effectively creating a dense point cloud that represents the entire scene.

The effectiveness of photogrammetry primarily depends on detecting and matching features/key points within the images. The aim is to identify key points corresponding to the same point in 3D space across distinct 2D images. Photogrammetric methods identify key points at places with notable color changes, typically indicative of well-textured and distinct surfaces. However, the challenge arises when dealing with weakly textured and uniformly colored surfaces, such as plain white walls, which are common in indoor environments. In such cases, while the reconstruction quality may not match that of outdoor environments with strongly textured surfaces, photogrammetry is still capable of identifying the crucial feature points. These key points often align with the boundaries of objects and architectural elements within the indoor environment. As a result, the reconstructed 3D key points provide valuable information about the location and dimensions of objects within the environment.

Photogrammetry is typically divided into two main categories: aerial photogrammetry, which employs cameras positioned in the air, and terrestrial photogrammetry, where cameras are either handheld or mounted on tripods. Specifically, close-range photogrammetry, associated with terrestrial photogrammetry, deals with object distances of roughly 200 meters or less. On the other hand, small-format aerial photogrammetry falls in between these two categories, as it combines the advantages of an aerial vantage point with the proximity to objects, resulting in high image detail capture [54].

Figure 2.9 depicts the results of laser scanning and photogrammetry of three bridges [55]. Laser scanning generally results in high accuracy and precision point clouds, typically in the range of 1-5 cm error on vertical surfaces, making them appropriate for DTs requiring precise measurements, such as architectural and archaeological documentation, as well as industrial inspection and monitoring.

The accuracy of a photogrammetric point cloud depends on various factors such as camera calibration, camera resolution, the number of images, and the vertical/horizontal overlap between images. In well-controlled conditions, photogrammetry can generate a point cloud at a competitive accuracy and density level with laser scanning [55]. Laser scanning is appropriate for low and mid-range object capturing and generally requires heavy and expensive equipment with a careful setup. Photogrammetry, however, is

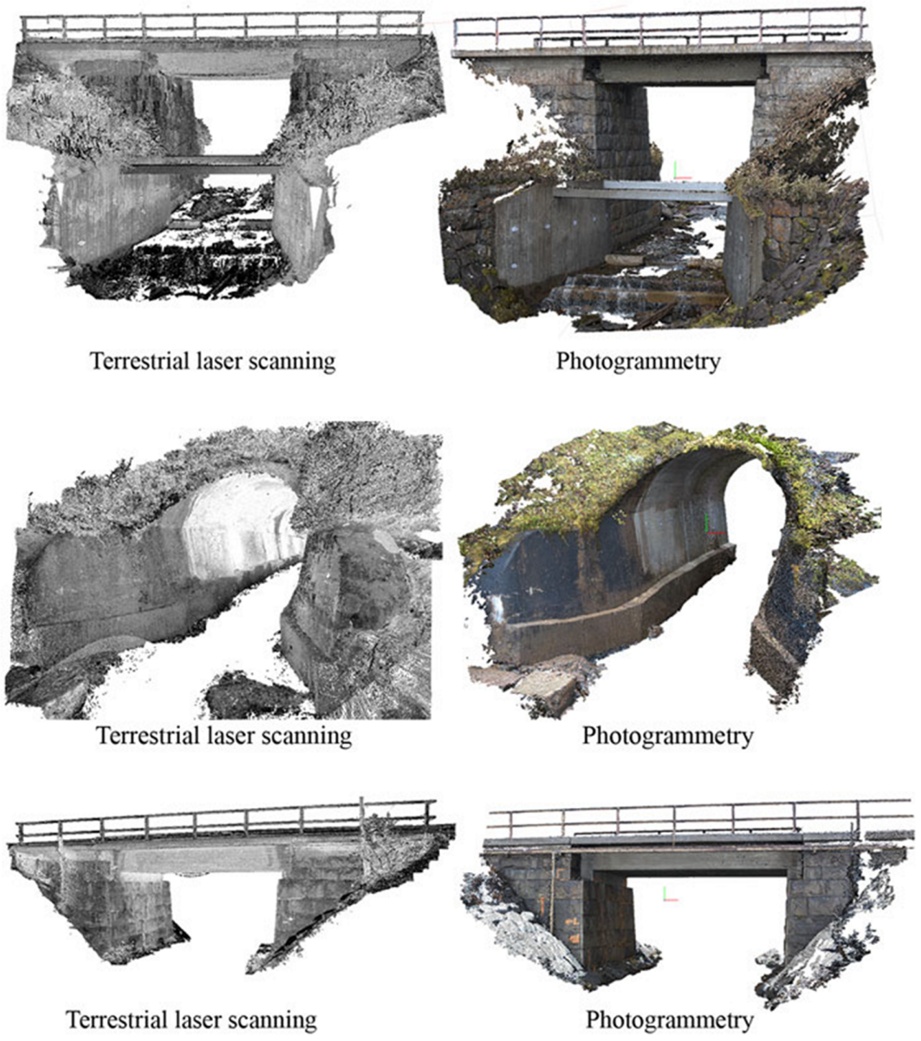

Terrestrial laser scanning
Photogrammetry
Terrestrial laser scanning
Photogrammetry
Terrestrial laser scanning
Photogrammetry

Figure 2.9: Visual comparison between laser scanning and photogrammetry [55].

scaleable, especially aerial photogrammetry, which paves the way for capturing hard-to-reach regions using UAVs. In terms of cost, photogrammetry can be more cost-effective, especially when using consumer-grade cameras and software [56]. A brief comparison of these two capturing methods can be seen in Table 2.4. The resulting point clouds from laser scanning and photogrammetry can be used in various applications such as surveying and mapping, architectural and engineering design, urban planning, environmental monitoring, etc. Point clouds can also be mentioned as the main data source for the geometric digital twining of bridges, as they represent the current geometric-semantic information of the structure.

Table 2.4: Comparison between Laser Scanning and Photogrammetry [57, 56].

	Photogrammetry	Laser Scanning
Modeling type	Image modeling	Range modeling
Data output	Point cloud data	Point cloud data
3D information	To be derived	Direct
Spatial resolution	Very high	high
Surface	Needs texture variation	Captures any type of surface
Scale	Absent	Present
Light	Dependent	independent
Intensity/Color	Very good	Limited
3D coordinate acquisition	Complicated	Direct
Accuracy and density	Medium to high	High
Range and coverage	Scaleable	Short to mid-range
Texture reconstruction	Very good	limited
Mobility and setup	Handheld devices and drones	Heavy equipments and careful setup
Time of data acquisition	Quite short	long
Instrument cost	Low	High

2.5 AI in the AECOM Industry

A large amount of data can be potentially generated throughout the lifecycle of assets with acquisition technologies and IoT devices, such as sensors, cameras, actuators, machines, etc. The collected data must be processed and interpreted efficiently to benefit from its potential advantages. AI is one of the most transformative and rapidly evolving fields in the realm of computer science and technology. It principally aims to create systems and machines that can mimic human cognitive processes, enabling them to reason, learn, perceive, and interact with the world. In the AECOM domain, AI provides the opportunity to convert data into actionable knowledge for a wide range of applications [58]. Machine Learning (ML), Neural Network (NN), Natural Language Processing (NLP), Computer Vision, and Optimization can be mentioned as sub-branches of AI with many applications in the AECOM industry.

2.5.1 Machine Learning (ML)

The field of ML focuses on developing algorithms and models that enable machines to learn and interpret input data to make predictions. ML employs computer systems to construct models that can subsequently facilitate decision-making and produce expected results for new inputs. The applications of ML encompass a diverse spectrum, including the automatic creation of knowledge representation models, the generation of knowledge bases for expert systems, planning, constructing numerical and qualitative models, text classification, text mining, knowledge acquisition for controlling dynamic processes, speech-to-text transcription, handwriting recognition, object recognition in images, and so on [59, 58]. Unsupervised, semi-supervised, and supervised learning are subbranches of ML with various levels of supervision to learn the input data. Supervised learning is devoted to the learning paradigms that need an annotation/labeling process before training, while semi-supervised learning needs a lower number of labeled data, and unsupervised can directly learn the input data without an annotation process.

2.5.2 Neural Network (NN)

Neural Network (NN)[1] form the major component of ML models. Inspired by the structure of neurons in human brains and strings, known as synapses, connecting the neurons, the mathematical model of NNs has been proposed. Figure 2.10 shows a simple NN model in which a set of neurons have been connected to each other with weighted links. Each neuron has a linear/non-linear function (activation function) that can generate predictive values from the input data (feature vectors). NNs create a system that is not only

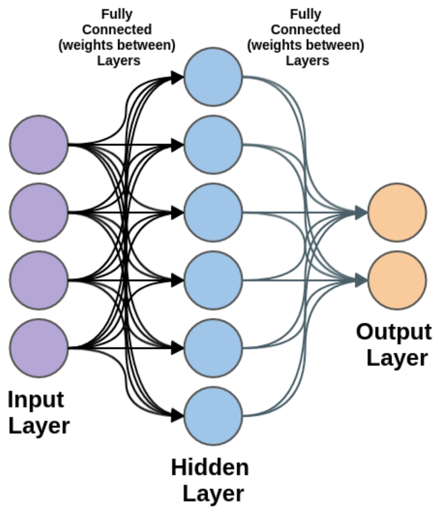

Fully Connected (weights between) Layers
Fully Connected (weights between) Layers
Input Layer
Hidden Layer
Output Layer

Figure 2.10: Multilayer Perceptron (MLP) with one hidden layer [60].

complex enough to describe complicated relationships but also derivable such that its weights can be tuned with gradient-based optimization algorithms in a backpropagation process. This mathematical model makes NNs capable of approximating trends among data points even in a high dimensional problem space that cannot be simply expressed

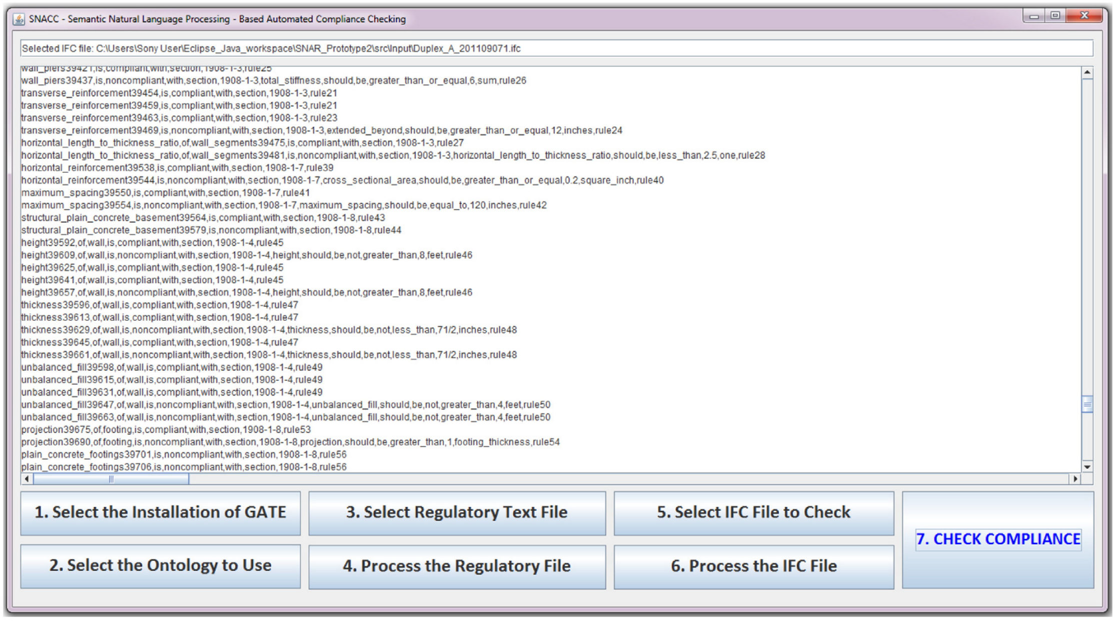
[1]Also denoted as "Artificial Neural Network (ANN)".

by conventional regression models. NNs have been used widely in various fields, such as NLP, computer vision, and robotics.

2.5.3 Natural Language Processing (NLP)

NLP is a field that deals with text as input to interpret, intercept, and generate human language. NNs are also the main component of NLP models to interpret text and generate desired results. NLP models generally need a preprocessing step known as tokenization to convert the input text to vectors representing the data. Various types of tokenization exist; the most common are word-level, character-level, and semantic-level. These methods only differ in how they assign a value to the input text. For instance, word-level tokenization assigns a specific number to each word, while character-level expresses each character by a number. Tokenization is generally followed by an embedding module in most NLP architectures. Embedding represents the tokenized text from a high-dimensional to a lower-dimensional space, allowing the network to learn more about the relationship between inputs and to process the data more efficiently. Text embedding modules mostly use a pre-trained model such as BERT (Bidirectional Encoder Representations from Transformers) [61] for precise input encoding. Text data is naturally sequential, i.e., a piece of text is a sequence of words that might have dependencies between them. To extract features from text and preserve these dependencies, NN modules such as 1D Convolution layers, Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU) can be employed [62, 63]. Most recently, there has been a significant surge in the adoption of Large Language Models (LLMs). These models use transformers that work with self-attention mechanisms. Self-attention enables the transformer to assign weights to different parts of a sentence such that the more important parts in a sequence can be emphasized in generating results. As a result, the number of weights decreases, and the model can learn quicker than the traditional models such as LSTM. The best example of an LLM is ChatGPT, which employs transformers to encode text.

The trace of NLP models can be seen in the AECOM domain. They have been employed to streamline and automate the process of evaluating compliance with building codes and regulations [64]. By parsing and analyzing textual documents, these models aid in ensuring that construction projects meet the necessary legal and safety requirements, reducing errors and expediting the compliance checking process [65]. These techniques have also been used to enrich textual data with structured information. This process involves extracting and categorizing data from unstructured texts, making it more accessible and useful for various AECOM applications [66]. This data enrichment facilitates better decision-making and improved data management. NLP-driven search engines and information retrieval systems enhance the accessibility of vast volumes of AECOM documents and data [67]. Operators can efficiently retrieve relevant information from project documentation, research materials, and historical data, saving time and ensuring that crucial insights are readily available. NLP models support the O&M of built environments. They support managing work orders, analyzing service requests, and generating maintenance schedules based on textual inputs, thereby enhancing the overall efficiency of FM processes. By extracting insights from textual data, construction managers can make informed decisions, monitor project milestones, and ensure timely project delivery [68]. A typical example of NLP-enabled systems is shown in Figure 2.11, where the system automatically extracts and transforms both regulatory information and design information in BIM for automated compliance reasoning.


SNACC - Semantic Natural Language Processing - Based Automated Compliance Checking
1. Select the Installation of GATE
2. Select the Ontology to Use
3. Select Regulatory Text File
4. Process the Regulatory File
5. Select IFC File to Check
6. Process the IFC File
7. CHECK COMPLIANCE

Selected IFC file: C:\Users\Sony User\Eclipse_Java_workspace\SNAR_Prototype2\src\Input\Duplex_A_201109071.ifc

```
wall_piers39421,is,compliant,with,section,1908-1-3,rule25
wall_piers39437,is,noncompliant,with,section,1908-1-3,total_stiffness,should,be,greater_than_or_equal,6,sum,rule26
transverse_reinforcement39454,is,compliant,with,section,1908-1-3,rule21
transverse_reinforcement39459,is,compliant,with,section,1908-1-3,rule21
transverse_reinforcement39463,is,compliant,with,section,1908-1-3,rule23
transverse_reinforcement39469,is,noncompliant,with,section,1908-1-3,extended_beyond,should,be,greater_than_or_equal,12,inches,rule24
horizontal_length_to_thickness_ratio,of,wall_segments39475,is,compliant,with,section,1908-1-3,rule27
horizontal_length_to_thickness_ratio,of,wall_segments39481,is,noncompliant,with,section,1908-1-3,horizontal_length_to_thickness_ratio,should,be,less_than,2.5,one,rule28
horizontal_reinforcement39538,is,compliant,with,section,1908-1-7,rule39
horizontal_reinforcement39544,is,noncompliant,with,section,1908-1-7,cross_sectional_area,should,be,greater_than_or_equal,0.2,square_inch,rule40
maximum_spacing39550,is,compliant,with,section,1908-1-7,rule41
maximum_spacing39554,is,noncompliant,with,section,1908-1-7,maximum_spacing,should,be,equal_to,120,inches,rule42
structural_plain_concrete_basement39564,is,compliant,with,section,1908-1-8,rule43
structural_plain_concrete_basement39579,is,noncompliant,with,section,1908-1-8,rule44
height39592,of,wall,is,compliant,with,section,1908-1-4,rule45
height39609,of,wall,is,noncompliant,with,section,1908-1-4,height,should,be,not,greater_than,8,feet,rule46
height39625,of,wall,is,compliant,with,section,1908-1-4,rule45
height39641,of,wall,is,compliant,with,section,1908-1-4,rule45
height39657,of,wall,is,noncompliant,with,section,1908-1-4,height,should,be,not,greater_than,8,feet,rule46
thickness39596,of,wall,is,compliant,with,section,1908-1-4,rule47
thickness39613,of,wall,is,compliant,with,section,1908-1-4,rule47
thickness39629,of,wall,is,noncompliant,with,section,1908-1-4,thickness,should,be,not,less_than,71/2,inches,rule48
thickness39645,of,wall,is,compliant,with,section,1908-1-4,rule47
thickness39661,of,wall,is,noncompliant,with,section,1908-1-4,thickness,should,be,not,less_than,71/2,inches,rule48
unbalanced_fill39598,of,wall,is,compliant,with,section,1908-1-4,rule49
unbalanced_fill39615,of,wall,is,compliant,with,section,1908-1-4,rule49
unbalanced_fill39631,of,wall,is,compliant,with,section,1908-1-4,rule49
unbalanced_fill39647,of,wall,is,noncompliant,with,section,1908-1-4,unbalanced_fill,should,be,not,greater_than,4,feet,rule50
unbalanced_fill39663,of,wall,is,noncompliant,with,section,1908-1-4,unbalanced_fill,should,be,not,greater_than,4,feet,rule50
projection39675,of,footing,is,compliant,with,section,1908-1-8,rule53
projection39690,of,footing,is,noncompliant,with,section,1908-1-8,projection,should,be,greater_than,1,footing_thickness,rule54
plain_concrete_footings39701,is,noncompliant,with,section,1908-1-8,rule56
plain_concrete_footings39706,is,noncompliant,with,section,1908-1-8,rule56
```

Figure 2.11: NLP-enabled system for automated compliance checking [68].

2.5.4 Computer Vision

Contrary to NLP, which mainly focuses on text, computer vision is a field concerned with interpreting visual data such as images, videos, and scans. The visual data required for computer vision can come from various sources such as cameras, satellite imagery, scanners, etc. Various tasks such as semantic segmentation, object detection, classification, motion analysis, image generation, and 3D reconstruction can be considered in this category. DL techniques, especially Convolutional Neural Network (CNN)s, have revolutionized NLP and computer vision. DL is a specific type of ML in which the model architecture generally contains a higher number of layers with complex dependencies for extracting features from the input data. As going forward in the encoding part of DL models, more features are extracted typically through convolution layers, thus increasing the depth of feature maps. Figure 2.12 shows the architecture of VGG 16 [69] in which the depth of feature maps increases while their size (width and height) decreases. The pyramid shape of the architecture can visually depict the definition of deep architectures.

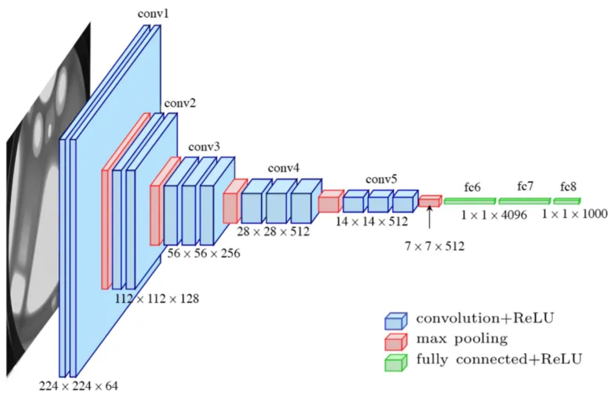


Figure 2.12: Architecture of VGG 16 [69].

From this model on, there have been fundamental architectures such as ResNet [70], GoogleNet [71], RCNN [72], YOLO [73], etc., each bringing new concepts and modules to the realm of ML and DL. For instance, in the ResNet architecture shown in Figure 2.13, the idea of using skip connections was proposed to address the vanishing gradient problem (the gradient in deep architectures becomes very small as we approach the initial layers; thus, the weights of those layers cannot be tuned properly).

Transfer learning is also a strong ML technique that has been widely used in computer vision. This approach uses a model previously trained on a dataset to tackle a related yet distinct task. Research studies have shown that the initial layers of DL models typically capture rough and fundamental features that can be transferred across other datasets, whereas the last layers capture fine and task-specific features. Hence, the weights of the initial layers can be frozen, and the model is only trained on its last/top layer(s). Fine-tuning in transfer learning is also a technique in which more layers from the top are unfrozen and tuned during the training process. This can improve the model's accuracy as it increases the adaptability of the pre-trained model for the new task.

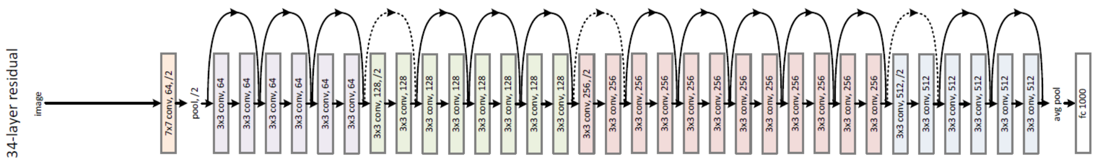


Figure 2.13: Architecture of ResNet [70].

DL architectures generally contain various modules/layers. Each module is employed for a specific operation depending on the expected tasks from the model. 1D, 2D, and 3D convolution, pooling, and fully connected layers are the fundamental building blocks for constructing DL network architectures. However, the ever-evolving field of DL constantly seeks to enhance the capacity of models using advanced modules to capture and process intricate patterns, information, and relationships within data.

Self-attention [74] is a module proposed for generating weights through fully connected layers and applying those weights to the feature vectors to emphasize the important features in generating final feature maps[2]. The proposed attention mechanism in the self-attention module has also found its way to other modules, such as Non-Local Attention (NLA) [75] and Global Context Block (GCB) [76], as shown in Figure 2.14.

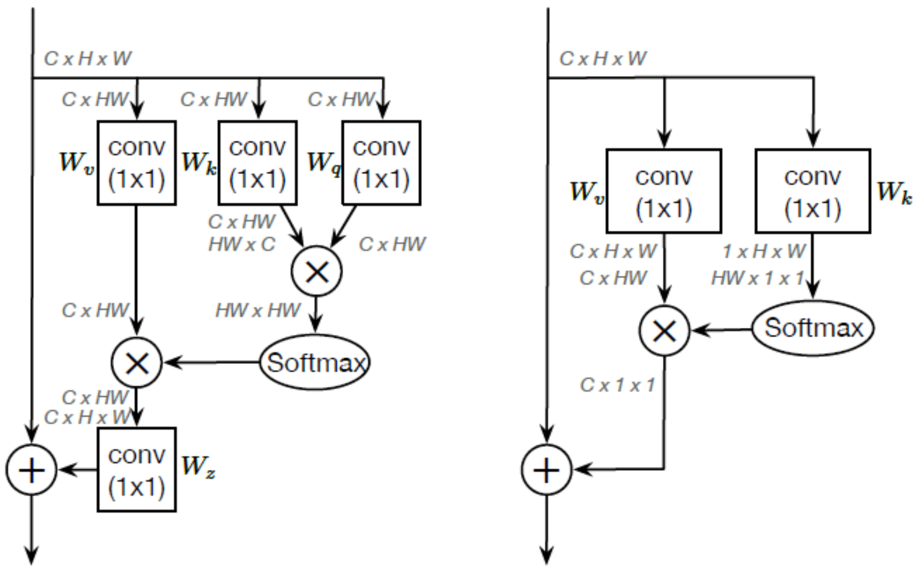


Figure 2.14: Spatial encoding of features using NLA (left) and GCB (right) [76].

[2]A feature map in a deep learning architecture is a three-dimensional array representing the output of neurons in a specific layer.

These modules also use a weighting mechanism to provide the spatial relationship among feature vectors that convolution layers cannot properly process. This is because the kernel size[3] in convolution layers is generally limited; thus, the pair-wise dependencies among pixels standing far from each other cannot be encoded efficiently. Contrary to the NLA module that uses two convolution layers to generate weights, GCB only employs a single convolution layer, leading to a lighter module that can be used in the sequential layers of the deep architectures.

For sequential data processing in tasks, such as motion analysis, NLP, and time series analysis, where each data point depends on its previous sample(s), specific types of NN modules, such as RNN, LSTM, and GRU (Figure 2.15), have been designed [62, 63]. RNNs contain recurrent cells that enable information to flow from one time step to the next. This recurrent connection forms a loop in the network, allowing the network to maintain a hidden state or memory of previous time steps. RNNs suffer from the vanishing gradient problem, making them unable to efficiently capture long-term dependencies. LSTM addresses this problem by introducing an input gate, a forget gate, and an output gate that can control the flow of new information into the cell, the data from the previous time step, and the data that should be read. GRU is also a simplified version of LSTM with only two gates (update and reset) that control the passed information from the previous step and data that should be forgotten [62].

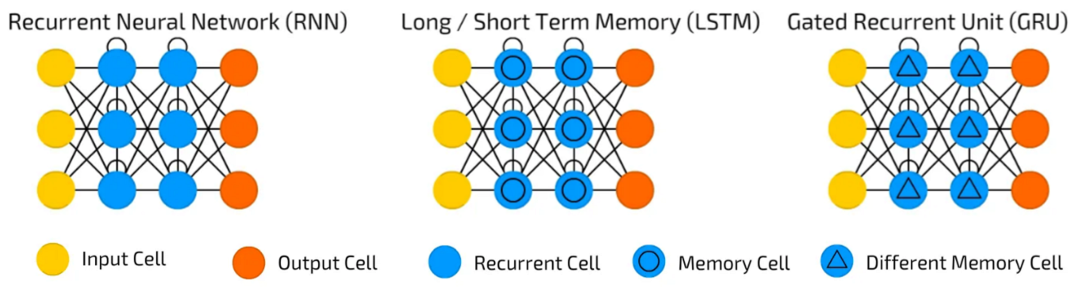

Recurrent Neural Network (RNN)
Long / Short Term Memory (LSTM)
Gated Recurrent Unit (GRU)
Input Cell
Output Cell
Recurrent Cell
Memory Cell
Different Memory Cell

Figure 2.15: Neural network modules for encoding sequential data [77].

Generative AI is also a subset of AI that has gained strong attention in recent years. Generative AI generates new data or content that resembles the input data it has been trained on. The input data can be text, image, audio, or video. Generative Adversarial Networks (GANs) [78] are fundamental architectures in the context of generative AI. GANs encompass two modules called generator and discriminator. These two modules always compete to produce results similar to the actual samples with some differences.

[3]The kernel size in a convolutional neural network represents the dimensions (height and width) of the filter or convolutional window applied to input data during the convolution operation.

The generator generates data using an input noise, while the discriminator is a binary classifier that tries to distinguish between the generated data and the real one (training sample). This adversarial process improves the quality of generated data by the generator. Most recent GAN architectures can not only generate data from the same data format but also from other formats. For instance, text data can be used to generate images corresponding to the content of the input text [79].

Computer vision also has a wide range of applications in the AECOM domain. Using the images and videos captured by cameras, these models can monitor the progress of the construction, ensuring the work is completed as scheduled. Defected or damaged areas of materials, structural elements, and welds can be detected automatically using computer vision models [80]. These models can also specify the type of corruption and recommend the next actions for facilitating the rehabilitation process [81, 82]. Computer vision models can monitor the construction site during the project's implementation phase to ensure safety compliance by detecting and altering potential hazards [83]. They can also observe the pose gestures of workers and analyze the surrounding environment to enhance safety measures [84]. These models can read and interpret technical drawings for automated 3D model reconstruction of buildings [85, 86]. This typical application can be seen in Figure 2.16, where building components such as walls, windows, and doors have been detected using computer vision models.

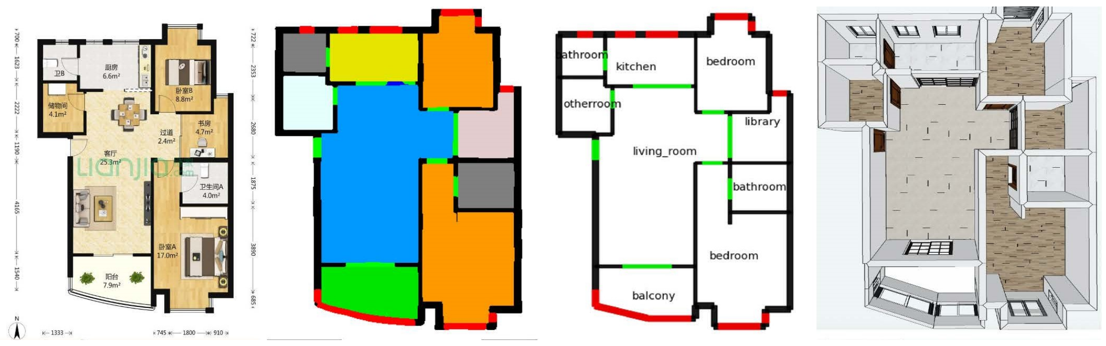


Figure 2.16: Model reconstruction of building floor plans using computer vision [85].

They can also compare the implemented project with the planned model and detect the discrepancies between these data sources, ensuring accuracy and adherence to architectural specifications [87]. Computer vision can assist in managing construction-related documents, such as blueprints, invoices, and contracts, by automatically scanning and classifying them [88]. GAN models can generate and recommend architectural plans based on a set of requirements such as the number of rooms, area, and dimensions [89]. They can also automate the model reconstruction process of buildings and bridges from scan data using 3D DL models. Augmented reality (AR) devices use computer vision to overlay digital information onto the real world, helping architects, engineers, and construction workers visualize designs and layouts in the physical environment [90]. Computer vision can optimize energy usage by monitoring real-time occupancy levels, lighting, and temperature conditions and adjusting accordingly.

2.5.5 Optimization

Optimization is a technique used to find the best/optimal solution in the feasible space of the problem. Optimization problems are generally expressed as minimization or maximization problems. The main goal of optimization algorithms is to minimize or maximize an objective/cost/loss function with respect to a set of parameters in a constrained or unconstrained problem space. The definition of the objective function is dependent on the problem. Based on the definition, various optimization algorithms can be used. For example, if the objective function and its constraints can be expressed in a linear format, linear programming methods such as simplex can be employed. Beyond this type of optimization, a diverse array of methods exists, each designed to address a category of optimization problems.

Most conventional optimization algorithms are deterministic, e.g., the simplex method in linear programming. Deterministic optimization algorithms that use gradient information are called gradient-based algorithms [91]. Gradient-based algorithms are a class of techniques used to find the optimal values of parameters in non-linear optimization problems. They are the type of first-order optimization requiring the first-order derivative to calculate the value of the gradient. These algorithms need a closed-form objective function to calculate the first derivative and guarantee convergence. They start from a point in problem space, and based on the value of the gradient and its direction, the algorithm updates parameters, as shown in Figure 2.17. It does so by stepping in the direction opposite the gradient. The step size is controlled by a parameter called learning rate. A smaller learning rate results in smaller steps, while a larger learning rate leads to larger steps. Even though gradient-based algorithms require a convex function, they can be used for non-convex problems with smooth problem space. An example of this can be seen in machine learning models that mostly use gradient-based algorithms such as Gradient Descent (GD), Stochastic Gradient Descent (SGD), Adam, and RMSprop in the training phase on non-convex space to minimize the loss value and tune the network weights.

Heuristic optimization, a problem-solving approach that prioritizes practicality over guaranteed optimality, utilizes methods to efficiently explore solution spaces. These

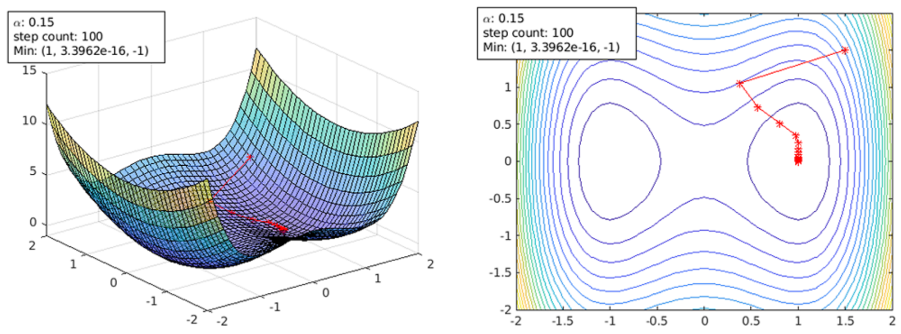


Figure 2.17: Convergence of a typical gradient-based algorithm [92].

approaches provide quick but approximate solutions, recognizing the trade-off between speed and the assurance of finding the global optimum. Metaheuristic computation is a sub-field of AI to solve optimization problems, especially with incomplete or imperfect data information [93]. The evolution of biological and natural systems has inspired most metaheuristic algorithms [94, 95]. Contrary to gradient-based optimization algorithms, metaheuristic algorithms are derivative-free and not dependent on the closed-form formulation of the objective function. This feature enables them to optimize nonlinear, multi-modal, and multivariate functions whose derivatives are not computable. Similar to other optimization techniques, metaheuristic algorithms require an objective/fitness function to evaluate the quality of the model. Most metaheuristic algorithms such as Particle Swarm Optimization (PSO) [96], Genetic Algorithm (GA) [97], Teaching Learning Based Optimization (TLBO)[98], Grey Wolf Optimizer (GWO) [99], and Firefly Algorithm (FA) [100] are population-based. This means a list of solutions/candidates is proposed initially based on the problem space (discrete or continuous) and the ranges of the parameters. This list is further improved by considering the fitness function value and the algorithm strategy, as shown in Figure 2.18. Finally, the best solution is reported as the global optimum location in the space of the problem.

Metaheuristic algorithms have also been used for energy, cost, and topology optimization of structures. In the context of energy management, metaheuristics have shown promising performance in optimizing the allocation and consumption of resources, leading to more sustainable and efficient energy utilization [102]. These algorithms enable intelligent decision-making processes, considering factors such as energy demand, cost constraints, and environmental impact. Topology optimization, another application area of metaheuristic algorithms, involves finding the optimal configuration of a structure or system to meet specific performance criteria. An example of these algorithms can be

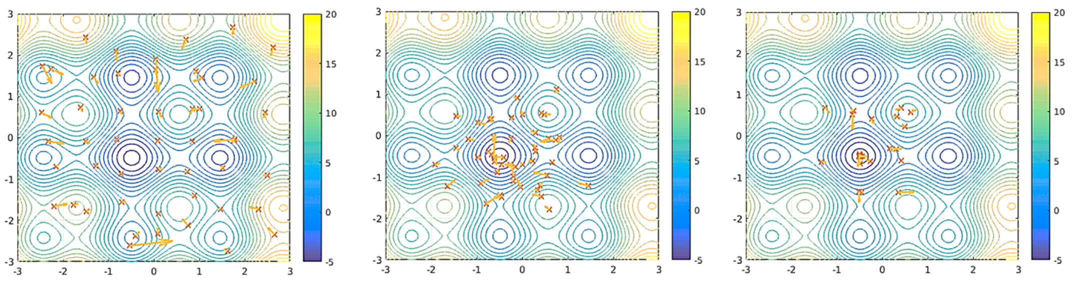


Figure 2.18: Convergence of a typical population-based metaheuristic algorithm as the number of iterations increases from left to right [101].

seen in truss topology optimization, where the members with higher contributions to sustaining loads are selected, as shown in Figure 2.19. Metaheuristic algorithms, with their ability to explore vast solution spaces and overcome local optima, play a significant role in achieving optimal designs. These techniques have been successfully employed in engineering industries to enhance components' structural integrity and performance [103]. In the context of NNs, the conventional gradient-based algorithm used for training the model can be replaced with metaheuristic algorithms, offering an alternative and complementary method for fine-tuning the weights and biases of NN. This technique improves the robustness and model performance, particularly when faced with non-convex and high-dimensional parameter spaces [94].

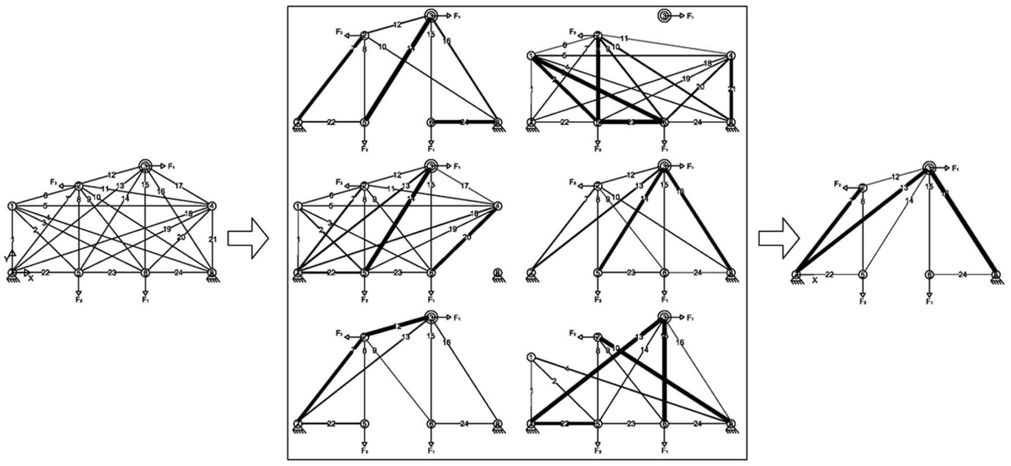


Figure 2.19: Truss topology optimization using metaheuristic algorithms [104].

2.6 Current Practice in Geometric Digital Twinning of Bridges

Despite the significant benefits of bridge DTs in the O&M phase of bridges, most transportation agencies still rely only on conventional systems to manage assets. Table 2.5 shows the results of a survey from engineers and contractors in 2017, where they were asked about the value a BIM model can provide in various phases of construction [105]. Across four countries of the US, UK, France, and Germany, most respondents believe that BIM delivers the most significant value during the design development phase. According to this report, only one of the 24 BIM-using owners has selected the maintenance phase. This observation implies that most transportation infrastructure owners currently view BIM as a tool to enhance design and construction processes without fully recognizing its potential for optimizing asset management and operational aspects.

Table 2.5: The results of a survey about the greatest value of BIM according to Engineers and Contractors (%) [105].

	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 Constructed (UK, France, Germany)	0	7	12	0
Maintenance (US)/Use (UK, France, Germany)	0	2	1	0

Another reason that prevents transportation agencies from broadly using DTs in the post-construction phase is the high cost and complexity of creating these models. At the core of a bridge DT, a geometric model exists that needs to be created in the initial step. This model can then be enriched with metadata collected by IoT devices such as sensors and is connected to a BMS. However, model creation is still the preceding problem in the digital twinning of bridges. Figure 2.20 shows the six required steps for generating a geometric DT, from point cloud generation to a 3D volumetric model. In what follows, a description of each step is brought up, and common practices are discussed.

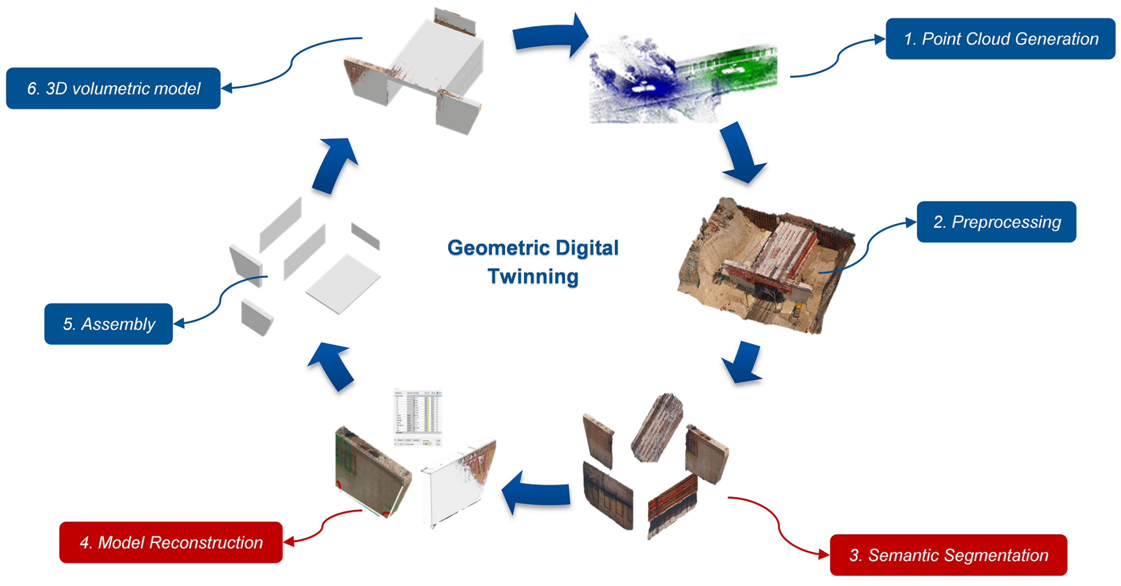


Figure 2.20: Required steps to create the geometric model of a bridge DT.

- Point cloud generation: As described in Section 2.4, PCD is the most appropriate data format for generating a geometric DT as it can represent the current geometric status of the structure. Creating point clouds typically involves advanced surveying and data acquisition techniques, such as laser scanning and photogrammetry, which are essential for developing an accurate and comprehensive digital twin. This step focuses on collecting precise and accurate data that will form the foundation of the geometric digital twin.

- Preprocessing: The output of laser scanning and photogrammetry is a raw PCD comprising millions or even billions of individual data points. This raw dataset, however, requires significant processing to be usable in creating a DT. This involves data alignment, noise reduction, and data registration, ensuring that all data points are accurately positioned in a common coordinate system and correcting errors or inaccuracies.

- Semantic segmentation: It involves identifying and classifying various elements within a given scene. When dealing with point clouds, such as those representing bridges, these points can represent various structural components, including railings, the bridge deck, abutments, and more. In many applications, such as model reconstruction, semantic modeling, and establishing spatial relationships among the components, it is highly advantageous to partition the entire point cloud of the bridge into smaller, more manageable subsets known as clusters or instances. This segmentation step offers two significant benefits. 1) By breaking down the entire point cloud into smaller clusters, the complexity of the task is reduced. This is particularly useful as these clusters typically consist of a lower number of points, making subsequent analysis more computationally efficient. 2) Each point cluster is assigned a label indicating the element type it represents within the bridge structure. This labeling is practical, as it provides clear and meaningful information about the identity and intent of each cluster. It essentially answers the question of which specific structural element every point cluster corresponds to.

- Model reconstruction: Point clouds generally consume a significant amount of memory as each point is stored as a separate data element. Also, point clouds cannot solely provide rich information about the geometric details of elements. Hence, they need to be processed further, enriched, and summarized to encompass the semantic-geometric information. Model reconstruction is a set of techniques to create or recreate a 3D model or representation of an object, scene, or environment. This process typically involves transforming real-world objects, scenes, or conceptual designs into digital 3D models that can be rendered, manipulated, and used in various applications, including animation, simulation, virtual reality, and Computer Aided Design (CAD). Triangulation/tesselation, mesh representation, boundary representation as well as procedural/parametric modeling are among the well-known model reconstruction techniques.

- Assembly: Piece-wise geometric modeling of bridge elements from their point clouds leads to a set of volumetric bodies that must be assembled for a unified and integrated 3D representation. In CAD software, assembling elements refers to positioning and connecting multiple parts or components to create a complete design or assembly. The components need to be interconnected correctly for a bridge to function as a cohesive structure. CAD software provides tools for specifying connections, welds, bolts, or other fasteners between elements. The software's parametric functionalities can be used to define relationships and dependencies between components, ensuring that changes in one part propagate correctly to related parts.

- 3D volumetric model: The last step is to check the integrity and compatibility of the model with the point cloud. This crucial phase serves as a final quality assurance process, ensuring that the 3D volumetric model accurately represents the real-world objects or environment from which the point cloud data has been derived. In this phase, the model's dimensions are cross-checked against the point cloud data to confirm that the scale is consistent. This is particularly vital for bridges where precise measurements are required. Furthermore, discrepancies or issues are detected and addressed through an iterative refinement process. This may involve modifying the 3D model, adjusting parameters, or making further measurements using the point cloud.

Among the aforementioned cases, *Semantic Segmentation* and *Model Reconstruction* are the most challenging steps requiring significant manual effort. Also, *Assembly* is a process that requires precise models for implementation; otherwise, neighboring elements cannot be integrated and might need an iterative process for the refinement of the generated models from the previous step. In current practice, various techniques and tools are employed to facilitate the geometric modeling of bridges. However, the process is still highly challenging, labor-intensive, and error-prone. To shed light on these challenges, the manual process of generating bridge models from point clouds is reviewed, and common methods and tools are described in more detail.

As mentioned, point clouds are generally bulky datasets containing millions of points. They, hence, require a large data storage space and processing resources. To address this problem, point clouds are generally subsampled. Subsampling of point clouds refers to the process of reducing the number of data points in a point cloud while attempting to preserve essential geometric or spatial characteristics. There are various methods for subsampling point clouds. Uniform Grid Sampling (UGS), Random Sampling (RS), and Furthest Point Sampling (FPS) are the most common sampling methods used in practice.

- Uniform Grid Sampling (UGS): It is a method of subsampling point clouds in which the space occupied by the point cloud is divided into a regular grid or lattice of equally sized cells or voxels. Each cell represents a small, uniform region within the point cloud's spatial extent. In UGS, one or more points from each cell are selected to represent that cell. This method helps reduce the density of the point cloud by keeping only a subset of the original points, typically one per cell, ensuring that the subsampled point cloud retains a uniform distribution of points across the entire dataset. UGS is often used to simplify PCD while preserving important spatial characteristics.

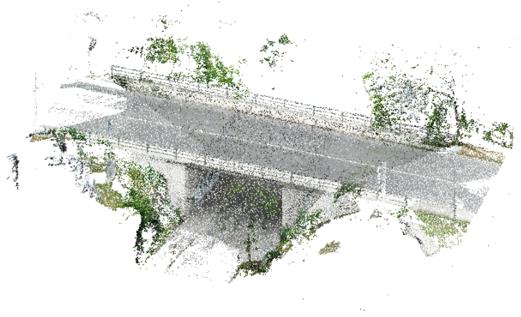
- Random Sampling (RS): It randomly selects a number of points from the original PCD. RS is the simplest and fastest existing subsampling algorithm. Contrary to UGS, which does not provide a mechanism for controlling the number of sampled points, RS can result in a specific number of points after subsampling. However, it may not preserve the spatial distribution or density of points in the original point cloud. It can result in areas with high point density being underrepresented in the sample while areas with lower density are overrepresented.

- Furthest Point Sampling (FPS): This technique aims to select points that are the furthest apart from each other, ensuring that the subsampled PCD retains the key spatial characteristics of the original dataset while reducing its size. FPS provides a mechanism to control the number of subsampled points and proper point cloud coverage. However, it is computationally more expensive than RS and UGS, limiting its applications. Figure 2.21 shows the results of applying each subsampling method to a bridge point cloud.

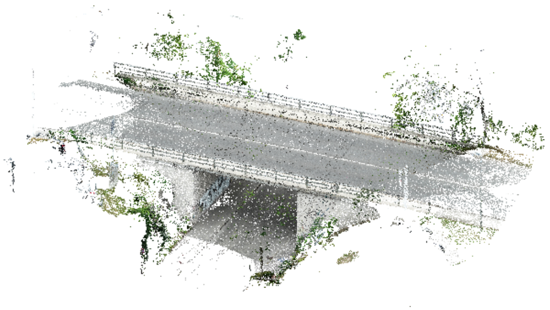
Resulting point clouds from laser scanning or photogrammetry are often corrupted by noise, which can arise from various sources, including sensor limitations, environmental factors, or the data acquisition process. To reduce the value of error from point cloud processing algorithms, raw point clouds are generally denoised.

Denoising is the process of removing or reducing noise while preserving the underlying structure and features of the 3D data. Filtering techniques such as median, Gaussian, or bilateral filters are typically used to remove noise. These filters can be applied to each point or neighborhood of points within the point cloud. Figure 2.22 shows a bridge point cloud after applying a Statistical Outlier Removal (SOR) filter. This filter calculates the average distance of points to their nearest neighbors in each local neighborhood and their standard deviation. It then removes the points that are farther than the average distance plus a number of times the standard deviation.

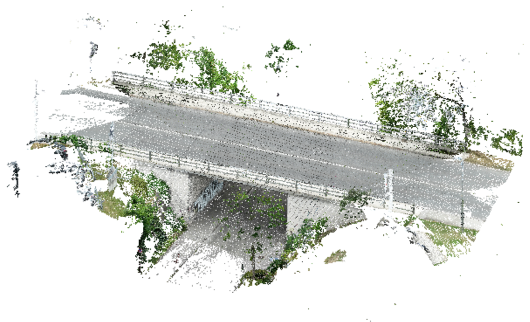
Transformation is the next preprocessing step performed to make the point cloud more manageable and easier to work. The transformation of point clouds is often referred to as Translation and Rotation. Translation is the process of moving/shifting the point cloud, while Rotation is defined as rotating the point cloud around the coordinate axes. Both cases can be performed by defining a transformation matrix and multiplying the point cloud with the matrix. In some cases, it is more suitable to translate the input point cloud to the origin of the coordinate system and rotate it around one or more coordinate axes.



(a)



(b)



(c)

Figure 2.21: Subsampled bridge point cloud using: a) UGS; b) RS; c) FPS.

Semantic segmentation is the next essential step to enrich the input point cloud. Semantic segmentation of point clouds is the process of dividing the input point cloud into the point cloud of various elements/components in the scene, as shown in Figure 2.23. In current practice, this step is often conducted manually by drawing polygons around the point cloud of elements belonging to the same family. Operators generally face several challenges during this step. Finding the best views for segmentation can be a time-consuming task. It involves rotating the point cloud to reveal the most infor-

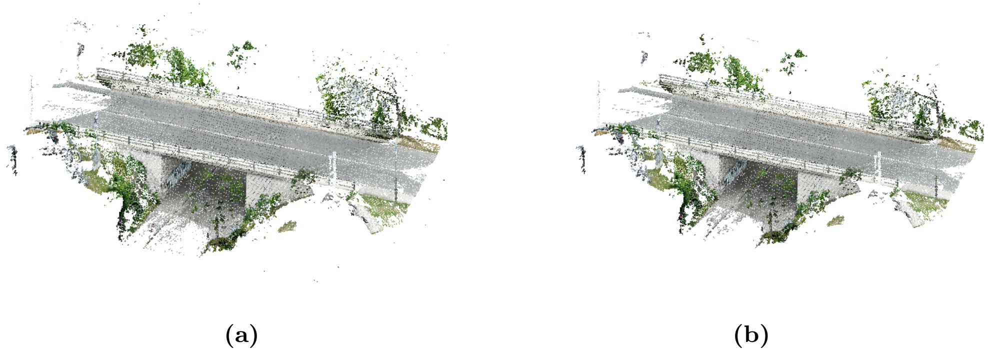


Figure 2.22: Point cloud of a bridge: (a) before de-noising; (b) after de-noising.

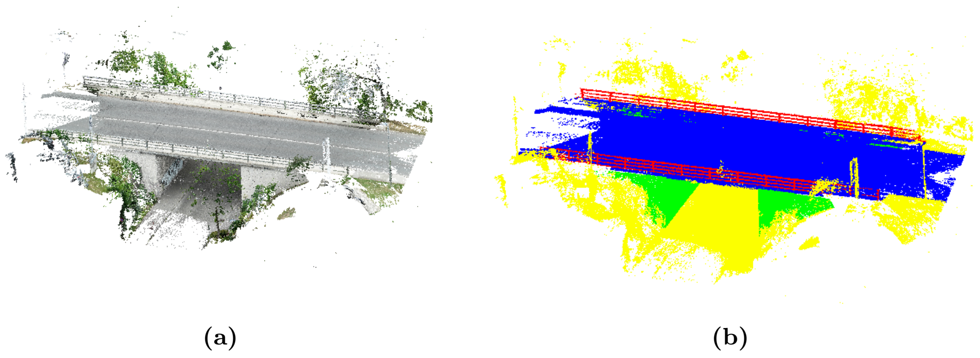


Figure 2.23: Semantic segmentation of a bridge point cloud: (a) before segmentation; (b) after segmentation. Class colors: yellow: background; blue: bridge deck; red: railings; green: abutments.

mative perspectives for separating points accurately. In complex scenes with multiple surfaces and intricate details, determining the optimal rotation angles and sequences can be labor-intensive. Furthermore, these selected views might still contain underlying points that are not observable due to occlusions or scanning limitations. Identifying and managing these unobservable points can pose a significant challenge during segmentation. Drawing polygons around point cloud segments is fundamental in creating point segments. However, it cannot provide a one-size-fits-all solution, especially when dealing with points not along straight paths or exhibiting complex geometries. Ensuring that the polygons accurately match the underlying structures can be a demanding task, requiring operator expertise. Each element within the point cloud usually requires at least one polygon for precise segmentation. This manual polygon drawing process can be both time-consuming and error-prone, especially in cases involving a large number of elements. Cropping each element by hand introduces the possibility of errors. Inaccurate cropping can result in incomplete or overlapping segments, which can negatively impact downstream applications that rely on the accuracy of the segmented data.

Semantic segmentation is generally followed by Instance Segmentation or Clustering. Instance segmentation aims to identify and distinguish individual objects or instances within a point cloud. Contrary to semantic segmentation, which groups points into categories (e.g., abutments, railings, bridge deck, background), instance segmentation assigns a unique label to each distinct object, even if they belong to the same category. For example, Figure 2.24 shows the results of instance segmentation in which the two instances of abutments and railings have been segmented further even though they belong to the same class.

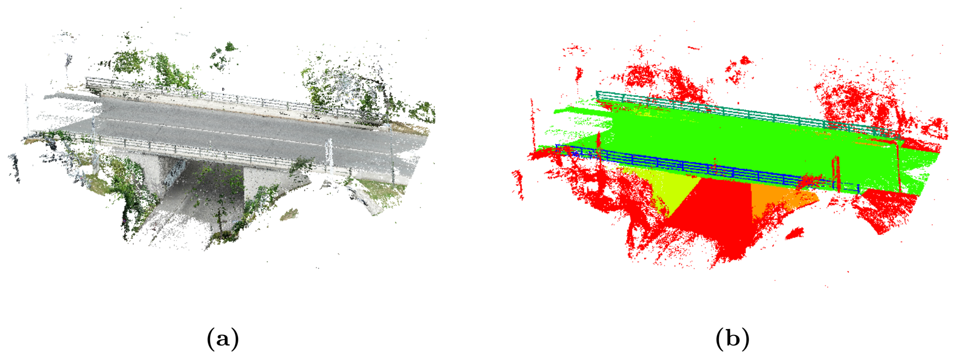


Figure 2.24: Instance segmentation of a bridge point cloud: (a) before segmentation; (b) after segmentation. Class colors: yellow and orange: abutment instances; blue and dark green: railing instances; red: background; green: bridge deck.

Once the segments are created, they must be saved and labeled individually. This data management aspect becomes highly challenging in larger projects where numerous segments must be organized, cataloged, and appropriately labeled for easy retrieval and analysis. Without efficient data management practices, the sheer volume of segmented parts can lead to confusion and inefficiencies. Furthermore, ensuring that the overall segmentation process remains standardized and consistent across the project is essential, especially in collaborative environments. Maintaining quality control and iterative adjustments to correct errors further adds to the complexity of the segmentation task.

To find access to the bridge's geometric details, each element needs to be modeled from its corresponding segmented point cloud. In current practice, two approaches are commonly used for the volumetric modeling of bridge components, as shown in Figure 2.25. The first approach measures the distances between points to estimate the parameter values corresponding to each bridge component. This comprehensive set of measured dimensions is then saved and labeled, ensuring they can be retrieved for further analysis

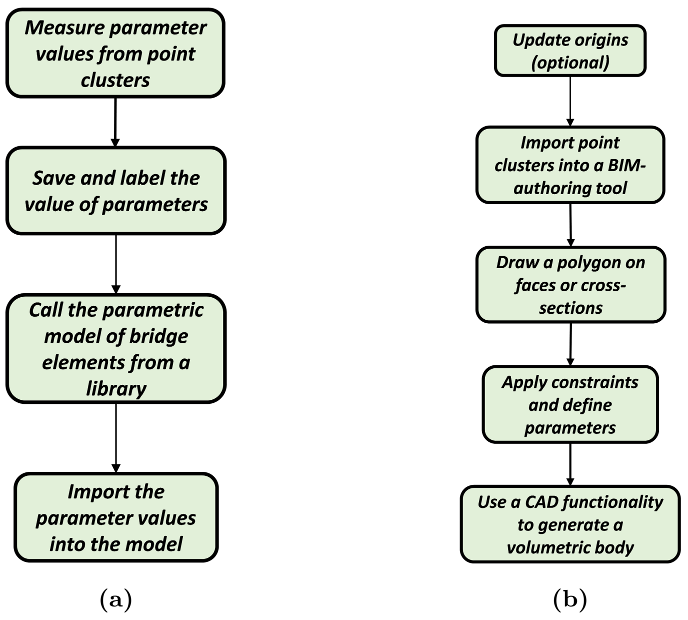
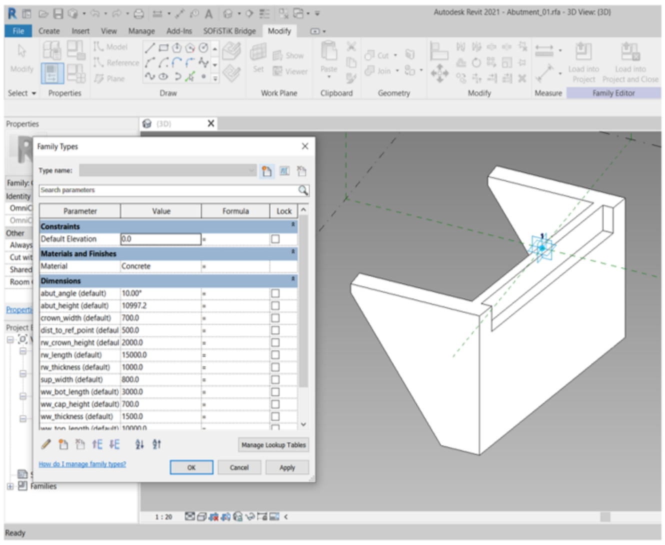

Measure parameter values from point clusters
Save and label the value of parameters
Call the parametric model of bridge elements from a library
Import the parameter values into the model
(a)
Update origins (optional)
Import point clusters into a BIM-authoring tool
Draw a polygon on faces or cross-sections
Apply constraints and define parameters
Use a CAD functionality to generate a volumetric body
(b)

Figure 2.25: Common approaches for geometric modeling of bridge components: (a) first approach; (b) second approach.

and manipulation. Simultaneously, a library of crafted dummy parametric models is created for each individual bridge component. These models are designed to be inherently adaptable, aligning with the parametric nature of the bridge components. They are created to enable the geometry of the dummy models to be dynamically altered in response to changes in the parameter values. In the next step, the previously saved parameter values are imported into these parametric dummy models, as shown in Figure 2.26. This integration enables the bridge component to dynamically adjust its shape. In doing so, it represents and reconstructs the segmented point cloud of the element. This approach combines measurement, model creation, and parametric adjustment to represent the bridge components.

While there are benefits in employing parametric models to facilitate the modeling process, the task of measuring every dimension within a point cluster can be laborious and time-consuming. Additionally, the process of saving and labeling each parameter value can be challenging, particularly in the case of bridges with many elements belonging to the same class; for example, assume a bridge with ten piers. Furthermore, there is the potential for inaccuracies in the derived parameter values from the point cluster, primarily due to the possibility of human errors. This potential for inaccuracies intensifies the need for precise attention to detail and rigorous quality control, as even a minor discrepancy in parameter values can have major effects on the overall structural



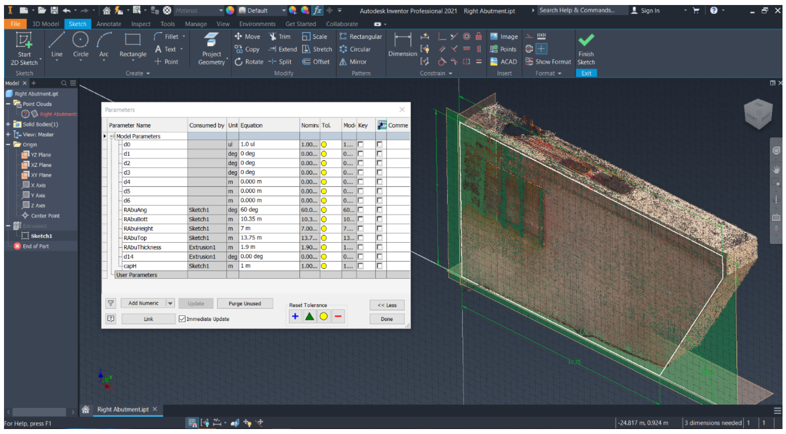
Parameter	Value	Formula	Lock
Constraints			
Default Elevation	0.0	=	☐
Materials and Finishes			
Material	Concrete	=	
Dimensions			
abut_angle (default)	10.00°	=	☐
abut_height (default)	10997.2	=	☐
crown_width (default)	700.0	=	☐
dist_to_ref_point (defaul	500.0	=	☐
rw_crown_height (defaul	2000.0	=	☐
rw_length (default)	15000.0	=	☐
rw_thickness (default)	1000.0	=	☐
sup_width (default)	800.0	=	☐
ww_bot_length (default)	3000.0	=	☐
ww_cap_height (default)	700.0	=	☐
ww_thickness (default)	1500.0	=	☐
ww_top_length (default)	10000.0	=	☐

Figure 2.26: Importing the value of parameters into an abutment instance.

integrity and performance of the bridge, emphasizing the critical role of precision and accuracy in the modeling and design of bridges.

In the second approach, the point clusters are directly imported into a BIM-authoring software, which serves as the platform for generating the 3D model. Prior to this step, a preprocessing procedure can be undertaken to reposition the origin of the point clusters, streamlining subsequent operations. This step ensures the selection of an appropriate reference or working plane aligned with the surfaces or cross-sections of the point clusters. The modeling process starts with drawing 2D sketches, often in the form of a polygon or other relevant shapes, directly on the selected reference plane, as shown in Figure 2.27. These initial sketches serve as the architectural blueprint for the forthcoming 3D model. Simultaneously, constraints such as parallelism, connectivity, perpendicularity, and symmetry are applied to ensure that the evolving design aligns with specific requirements and standards. If parametric modeling is of interest, functions controlling the geometry are defined as well, affording the model the flexibility to adapt and respond to evolving design criteria. These 2D sketches are finally transformed into a volumetric body using a CAD functionality such as Extrude, Sweep, Revolve, or Loft.

In direct modeling using point clusters without relying on parametric models, it is essential to recognize that this approach necessitates a repetitive geometric modeling procedure that must be rigorously conducted for each point cluster. The iterative nature of this process requires more manual effort when confronted with the substantial volume of point clusters associated with bridge structures. Consequently, what may


Parameters

Parameter Name	Consumed by	Unit	Equation	Nomina	Tol.	Mod	Key		Comme
Model Parameters									
d0		ul	1.0 ul	1.00...	●	1....	☐	☐	
d1		deg	0 deg	0.00...	●	0....	☐	☐	
d2		deg	0 deg	0.00...	●	0....	☐	☐	
d3		deg	0 deg	0.00...	●	0....	☐	☐	
d4		m	0.000 m	0.00...	●	0....	☐	☐	
d5		m	0.000 m	0.00...	●	0....	☐	☐	
d6		m	0.000 m	0.00...	●	0....	☐	☐	
RAbuAng	Sketch1	deg	60 deg	60.0...	●	60...	☐	☐	
RAbuBott	Sketch1	m	10.35 m	10.3...	●	10...	☐	☐	
RAbuHeight	Sketch1	m	7 m	7.00...	●	7....	☐	☐	
RAbuTop	Sketch1	m	13.75 m	13.7...	●	13...	☐	☐	
RAbuThickness	Extrusion1	m	1.9 m	1.90...	●	1....	☐	☐	
d14	Extrusion1	deg	0.00 deg	0.00...	●	0....	☐	☐	
capH	Sketch1	m	1 m	1.00...	●	1....	☐	☐	
User Parameters									

Figure 2.27: Geoemtric modeling of a wing wall in a BIM-authoring software.

initially seem a streamlined method can rapidly evolve into a challenge due to the extensive data and details that must be addressed. Also, a notable challenge emerges in the precise definition of reference and working planes for creating 2D sketches. The limited availability of straightforward and intuitive tools within most BIM-authoring software platforms further compounds this challenge. Defining accurate reference planes without such tools necessitates a profound understanding of the software's capabilities and a significant degree of manual input. Therefore, an additional layer of complexity is added to the modeling process, underscoring the requisite proficiency and meticulous attention to detail for achieving accurate and consistent results. This is particularly critical when dealing with the complex geometric characteristics of bridges. Furthermore, inaccuracies in drawing 2D sketches are still a potential risk that might end up with major inconsistency in modeling the bridge components.

The piece-wise modeling process of bridge components leads to the volumetric model of each element. These elements must be assembled to form a cohesive and integrated model that accurately represents the entire bridge. To this end, the individual elements are imported into an assembly environment, wherein their interrelationships and dependencies can be defined through a set of constraints. The assembly process requires precise geometric models of elements, ensuring they can be interconnected and fit together properly. For instance, when considering two adjacent elements, such as a wing wall and a retaining wall, it is reasonable to expect they share a common edge with the same height. However, it is common that the generated models from the modeling step cannot meet all the requirements due to the inherent potential for human errors that may occur during the modeling process from point clouds. The assembly process might not be conducted properly if discrepancies and inconsistencies exist, such as variations in the shared edge height between the wing and retaining walls. In such cases, the assembly process necessitates a refinement phase to rectify these inconsistencies.

It is essential to emphasize that the quality assurance procedures employed in manual modeling primarily rely on visual comparisons between the generated model and the underlying bridge point cloud. However, a significant limitation is that most existing software tools lack mechanisms for quantifying the modeling accuracy. A notable challenge in this regard is the absence of a clearly defined and quantifiable metric for evaluating the degree of similarity or proximity between the resulting geometric model and the point cloud. While visual evaluation can be a valuable qualitative assessment method, the absence of a precise quantitative measure poses challenges in achieving a thorough and accurate assessment of the model.

3 State-of-the-Art

Despite the considerable advantages of DT models in facilitating the O&M process of bridges, the costs associated with manual DT modeling currently outweigh the short-term benefits from the model. To achieve an efficient method for bridge DTs, the costs associated with digital twining must be reduced to a large extent.

At the core of a bridge DT a geometric model exists whose creation is, however, costly, labor-intensive, and error-prone. Among all the steps required for the geometric digital twinning of bridges, *semantic segmentation* and *model reconstruction* are essential yet the most challenging ones. In current practice, these steps are conducted manually, increasing the cost and time concerning digital twinning. Considering the large number of bridges and the high demand for providing an efficient mechanism for bridge management and quality assessment, those steps need to be automated or at least semi-automated. In recent years, there have been efforts to automate the geometric modeling of bridges from PCD.

This section presents an overview of the concepts, methods, and techniques proposed for the automation of the PCD-to-DT process. Furthermore, it highlights the existing research gaps and the key contributions of this research.

3.1 Semantic Segmentation of Bridge PCD

Semantic segmentation of point clouds is a 3D data processing task that assigns a semantic label or category to each individual point in a point cloud. In computer vision, it is defined as a classification at the point level as it aims to classify each point in the cloud into predefined categories or classes [106]. Semantic segmentation has various applications in identifying and categorizing objects and structures within a 3D environment for autonomous vehicles, robotics, urban planning, and augmented reality [107, 108, 109]. It also enhances the understanding of the 3D scene by providing a detailed representation of its components, enabling more advanced decision-making processes.

Semantic segmentation of bridge point clouds is a crucial process that not only divides the dataset into distinct segments representing bridge elements but also enriches the point cloud by introducing a new attribute, namely class labels. By classifying points, semantic segmentation supports various tasks such as model reconstruction, structural analysis, planning, etc. Various sets of features, such as (x, y, z) coordinate of points, RGB color codes, normals, and so on, can be used for point cloud segmentation. However, most existing models, algorithms, and techniques, at least, use the (x, y, z) coordinate of points as the input data. The level of semantic segmentation is completely user-dependent and is defined based on the requirements and expected use cases. For instance, a bridge point cloud might be segmented into the point clouds of sub- and super-structure, while in another case, it might be segmented further to the point cloud of elements such as railings, bridge deck, piers, abutments, etc.

Bottom-up, *top-down*, and *deep learning-based* are common approaches widely employed in the semantic segmentation of bridge point clouds. Each approach has distinct advantages and drawbacks depending on several factors, including the required number of thresholds, the degree of automation, efficiency against occlusion and noise, processing efficiency concerning speed and memory, and the scope for adaptability when applied to previously unseen datasets. A robust method/algorithm for semantic segmentation of bridge point clouds is the one with the lowest number of sensitive thresholds, as the calibration of these values directly impacts the level of automation. Furthermore, it provides a high level of automation by minimizing user intervention in processing point clouds. In addition, it must demonstrate acceptable performance in scenarios characterized by visual and physical occlusion as well as noise, all of which are common challenges encountered when dealing with point clouds. The solution should also strive for a certain level of generality, capable of performing adequately when confronted with unseen datasets. Additionally, it should be capable of processing large volumes of data within reasonable time frames, as the input point clouds may encompass millions of data points.

This section reviews three common approaches for the semantic segmentation of bridge point clouds and brings the related works into the scope of each method. It further presents the advantages and drawbacks of the methods, aiming to provide readers with deeper insights into their respective merits and limitations.

3.1.1 Bottom-up Semantic Segmentation

The bottom-up methods transform the low-level features into a set of high-level features and leverage them to generate a more complex system at successively higher levels [110]. The low-level features are generally the raw attributes of PCD, such as the x, y,

z coordinates of points, and the RGB color codes. The high-level features are typically the surface normal, meshes, surface planes/patches, non-uniform B-Spline surfaces, and voxels [111, 112, 113, 114, 115, 116]. Multiple algorithms are capable of utilizing these features for primitive detection and model reconstruction. Region growing (RG), RANdom SAmple Consensus (RANSAC), and Hough-Transform (HT) can be mentioned as dominant algorithms in this category. Most of these algorithms have been proposed initially for the segmentation of images [117, 118, 119] and have been extended to 2D/3D point clouds.

Region Growing (RG) is the most well-known category of bottom-up methods. These algorithms start from one or more seed points (e.x., randomly selected points) and expand the region to cover objects of interest in a point cloud [120, 121]. RG algorithms generally employ a data structure such as kd-tree or octree and a search algorithm such as K-nearest neighbors search (KNN) or range/ball search to detect the nearest neighbors of points. They then start from the seed points and employ a set of features (low-level or high-level) to grow/expand the region [122, 123]. For instance, when detecting a plane, point normals can be used. This step is generally controlled using a set of thresholds so that the points with similar features are added to the same cluster; for example, the deviation in the angle of normal vectors is checked in the plane detection task. The RG algorithms stop adding points to a cluster when no point is left in their neighborhoods with similar features, i.e., the points whose features meet the predefined conditions. The algorithm finishes when a label is assigned to each individual point in the point cloud.

RANdom SAmple Consensus (RANSAC) is an algorithm based on sampling an object in a scene and evaluating its similarity using a set of thresholds [119]. In the case of point clouds, the RANSAC algorithm can also be considered a bottom-up method as it samples objects using a minimal number of seed points. For instance, a plane is sampled with three randomly selected points. The similarity of the sampled object is quantified using an objective function. This function calculates the object's distance to the points and classifies points as inliers if they are placed within a predefined distance/bound to the sampled shape [124]. For example, the distance of points to a randomly sampled plane is measured, and the points within a 10 cm distance to the plane are counted as inliers. Over a number of iterations, RANSAC selects the sampled object with the highest number of inliers as it shows a higher similarity to the desired object/shape. RANSAC is an iterative algorithm that requires an objective function to measure distances and detect inlier points. Therefore, the RANSAC algorithm has been defined mainly for primitive shapes such as Planes, Spheres, Cylinders, Cones, and Torus whose objective functions can be defined in a closed-form formulation and evaluated promptly [124].

RANSAC is highly robust and can detect objects even in noisy conditions; however, it needs a set of thresholds and hyperparameters that might not be easy to determine [125].

Hough Transform (HT) is a technique used primarily to detect primitive shapes, such as lines, circles, and ellipses, within images and 2D/3D point clouds. It was initially proposed as a technique for image processing [126]. The HT algorithm starts with representing geometric features in a specific parameter space. For example, for the detection of lines, each data point is represented as parameters describing a line in the parameter space, e.g., slope and intercept of the line. Parameter space represents an n-dimensional space in which each data point corresponds to a combination of parameters describing the geometric shape. The parameter space of a line can be a two-dimensional space with the axes slope and intercept; each data point in this space represents a line. HT algorithm creates a voting system for every data point based on a set of thresholds describing the similarity of the data to the target shape. The voting process is repeated for each data point, and the results are accumulated. After all votes are cast, the data points with the highest votes are detected, as these points are more likely to represent the object.

Bottom-up algorithms have been used principally or partially in detecting building and bridge components. Murali et al. [127] detected walls in building point clouds and employed subsequent reasoning about the spatial relationships to generate the 3D BIM model with individual rooms refined by walls, ceilings, floors, and doors. Boulaassal et al. [128] employed RANSAC to extract the planar surfaces of facades such as walls and windows. In another study, Hulik et al. [129] recognized planar surfaces in the building point clouds using a 3D HT algorithm, showing that it can achieve competitive results with the RANSAC algorithm in a well-controlled condition. Lee et al. [130] detected planes in a bridge point cloud using M-estimator SAmple Consensus (MSAC), a generalization of the RANSAC [131], and extracted the value of parameters for specific types of the bridge deck by measuring the distance between each pair of planes. Yan and Hajjar [132] used RANSAC to detect planes of steel girders in the bridge point clouds and reconstructed the steel sections after refining the planes. Chen et al. [133] detected a bridge deck point cloud using an RG algorithm after calculating high-level features such as normals, RGB color codes, local density, roughness, and Gaussian curvature. An example of using the RG algorithm for semantic segmentation of bridge point clouds can be seen in [134]. This study employed a Quadtree (Octree in 2D) as a data structure and a cell- or voxel-based region growing (CRG/VRG) algorithm, as shown in Figure 3.1, to extract the planar surfaces in the PCD of RC bridges.

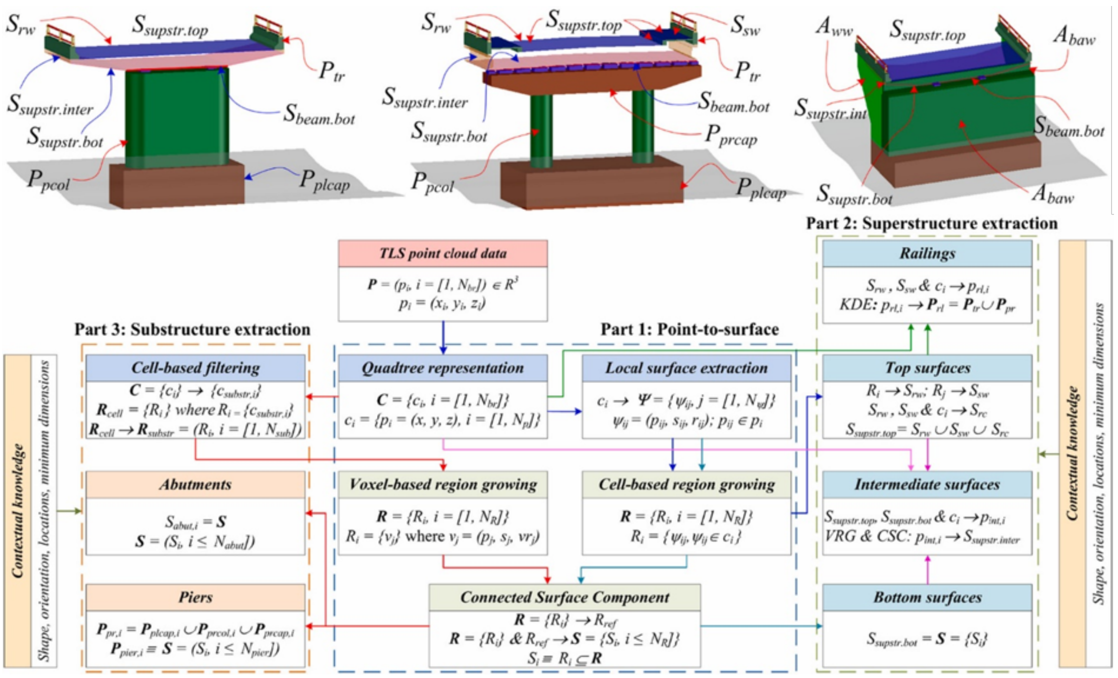


Figure 3.1: A bottom-up method for semantic segmentation of bridge point clouds [134].

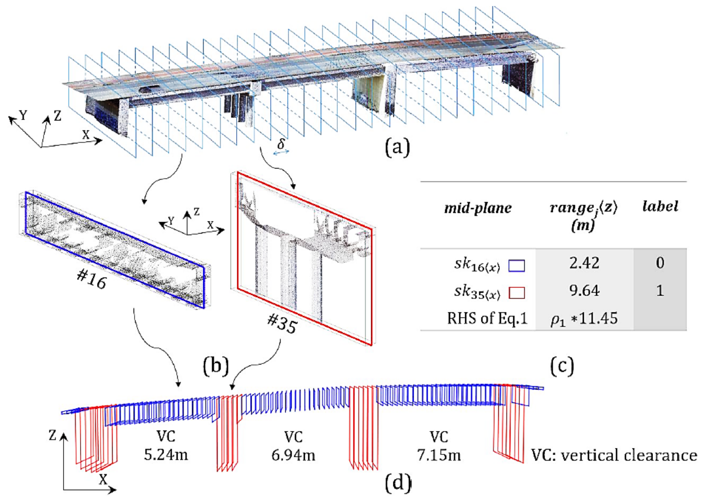
There are fundamental algorithms with various applications in the category of bottom-up approaches. These algorithms can be used to detect elements in a point cloud scene. However, bottom-up algorithms have been primarily devoted to generating surface-based geometries, especially planar surfaces and primitive shapes such as the sphere, cylinder, cone, and torus [124, 135]. This is mainly due to the dependency of these algorithms on a closed-form formulation for the fast evaluation of the sampled geometry. Bridge elements generally have more complicated geometric shapes than that of primitive shapes. Hence, bottom-up algorithms often need other post-processing algorithms and might be unable to solely detect all bridge components in a point cloud. Furthermore, they primarily require a set of thresholds, while tuning them might not be simple, especially in point clouds with different resolutions. Last, they generally suffer from occlusion and clutter, which is common in bridge point clouds.

3.1.2 Top-down Semantic Segmentation

Contrary to bottom-up, top-down methods start from a complex system and decompose it to subordinate systems or elements that are simpler to interpret [136]. They require hierarchical planning and deep insight into the system so that coding cannot be started without reaching a sufficient level of detail, at least for some parts. One prominent feature of top-down algorithms is in the divide-and-conquer application. They divide a complex problem into smaller parts, solve each part individually, and then combine solutions to address the overall problem. This approach has been commonly used in sorting algorithms such as merge sort and quicksort [137]. The top-down methods are often heuristic established based on a set of domain-specific information/criteria. This knowledge-based approach can leverage the existing information, such as predefined constraints, parameters, object types/instances, and topological relationships, to break down the initial complex model to a set of basic elements [138, 139, 140, 141]. These basic elements can be further refined in more detail and in many additional subsystem levels. A pioneering study using a top-down approach was REFAB (Reverse Engineering FeAture-Based)[142], which uses parametric geometric constraints such as parallelism, connectivity, perpendicularity, and symmetry to convert points to mechanical models. The top-down approach has also been used for the semantic segmentation of bridge point clouds.

To overcome the limitations of bottom-up segmentation algorithms, Lu et al. [143] proposed a heuristic top-down method for detecting bridge components following directional geometric cuts and measuring the relative distance of points, as shown in Figure 3.2. Zhao et al. [144] defined a heuristic algorithm using the horizontal histogram of density and a support vector machine (SVM) for portioning structural elements in RC bridges. Yan and Hajjar [145] proposed a top-down heuristic method for semantic segmentation of steel girder bridges through the existing geometric and topological constraints in those bridges. Pan et al. [146] described a top-down method based on graph construction and combined it with a rule-based classification algorithm to segment the main elements of heritage bridges. Qin et al. [147] employed the local and global density of points for semantic segmentation of bridge PCD and fitted cylindrical and cuboid shapes to the segmented point cloud of elements. Qin et al. [147] used the local and global density of points for semantic segmentation of bridge PCD and fitted cylindrical and cuboid shapes to the segmented point cloud of elements.

Top-down algorithms might be less dependent on thresholds or hyperparameters than bottom-up methods. They might also show better performance in point clouds with occlusion and clutter. However, most top-down algorithms are defined considering a set of presumptions about the target bridge's geometric conditions or topological rules. Therefore, top-down algorithms might fail in cases where the input point cloud does not satisfy the presumed assumptions. For instance, a top-down algorithm developed for straight bridges might fail if the input sample has even small degrees of horizontal/vertical curvature. Furthermore, top-down algorithms mostly tend to provide explicit solutions



mid-plane	*range$_j\langle z\rangle$ (m)*	*label*
$sk_{16\langle x\rangle}$ 🟦	2.42	0
$sk_{35\langle x\rangle}$ 🟥	9.64	1
RHS of Eq.1	ρ_1 *11.45	

Figure 3.2: Top-down semantic segmentation of bridge point clouds: (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 [143].

through a deep insight into the problem components. However, finding and expressing such a solution might not be feasible in many cases.

3.1.3 Deep Learning-based Semantic Segmentation

Following the breakthrough results of applying DL models to images, there has been a strong interest in adapting such models to 3D geometric data such as meshes or point clouds. Contrary to images, a raw point cloud is an unstructured dataset without an underlying grid. Thus, its features cannot be extracted simply by passing 2D convolution and pooling layers [148]. To address this issue, the initial DL models mostly relied on transforming input point clouds to either multi-view images or volumetric data (Figure 3.3) that are readable by 2D and 3D convolutions [149, 150, 151]. However, the complexity and memory issues arising from the intermediate representations restricted the capability of these models.

PointNet [152] is the first 3D DL model with the ability to process and extract features from points directly. The architecture of PointNet is a sequence of shared MLPs with T-Net modules, as shown in Figure 3.4. Shared MLPs are 1x1 convolution layers that act similarly to fully connected layers. The T-Net module consists of a transformation matrix whose entries are calculated through the network. This matrix makes the model invariant to rigid transformations such as rotation and translation. PointNet also uses symmetric functions such as *max* or *mean* to achieve permutation stability and conquer

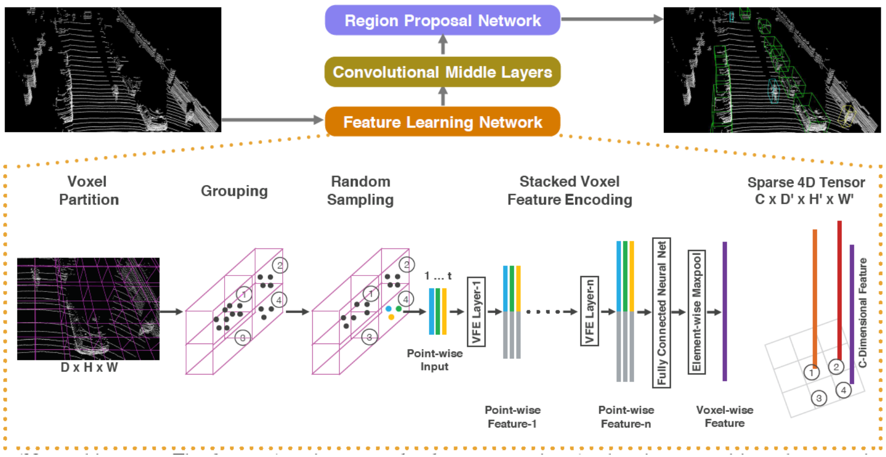


Figure 3.3: Architecture of VoxelNet [149]. Point clouds are transformed into voxels, and convolution layers are applied.

the unordered point cloud input. PointNet only uses the global features of points, such as (x, y, z) coordinates and RGB color codes for the classification of points. This means it does not provide information about the local neighborhoods and processes every point separately, considering its global features. It also cannot simply process large-scale point clouds in which the number of points might vary largely.

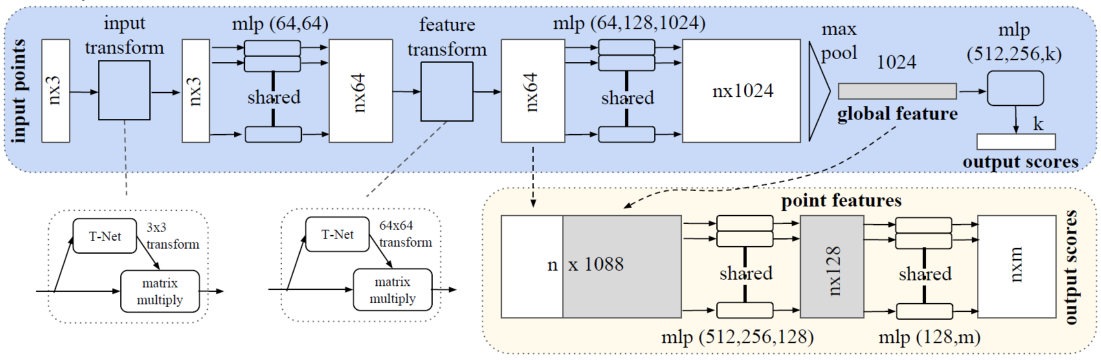


Figure 3.4: Architecture of PointNet in which global features are encoded using shared MLPs [152].

To address these issues, PointNet++ [153] was proposed that uses an FPS subsampling strategy in the sequential layers of the network and calculates the local features in neighborhoods obtained by a range/ball search algorithm, as shown in Figure 3.5. For the semantic segmentation task, the architecture of PointNet++ also converges to a U-net [154] architecture with an encoder, bottleneck, and decoder (autoencoder), as well as concatenation through skip links.

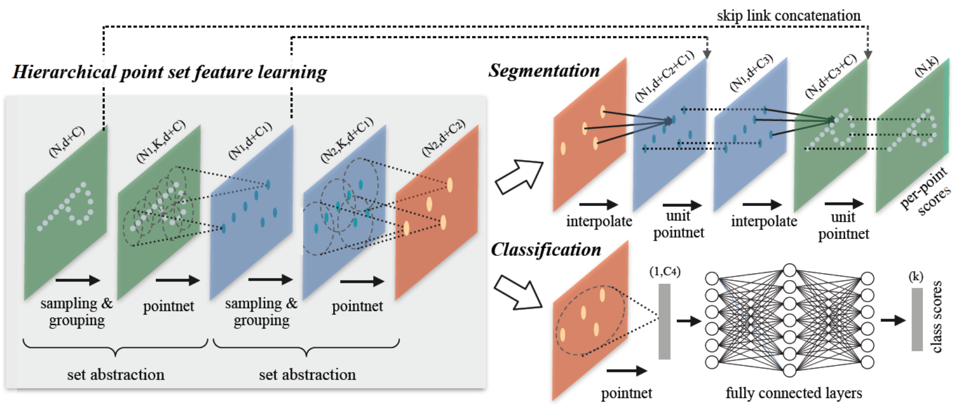


Figure 3.5: Architecture of PointNet++ [153] in which local features have also been encoded.

From this model onward, 3D DL architectures have been developed in three parts of the sampling strategy, feature extraction, and encoding-decoding process [155, 153, 156, 157]. For instance, RandLA-Net [158] employs RS as a more efficient sampling strategy for fast processing points. KPConv [157] proposes kernel point convolution modules/layers to extract the local features of points, while RepSurf [159] calculates the triangular and umbrella surface representations to encode the local neighborhoods, as shown in Figure 3.6. Thanks to the sub-sampling and up-sampling in subsequent layers of the U-shaped autoencoder, the DL architectures are capable of inferring per-point semantics of large-scale point clouds [158]. The encoder of these models has also been used for real-time semantic segmentation and model-to-cloud fitting of primitive shapes [160]. DL models have also been used in the semantic segmentation of bridge point clouds.

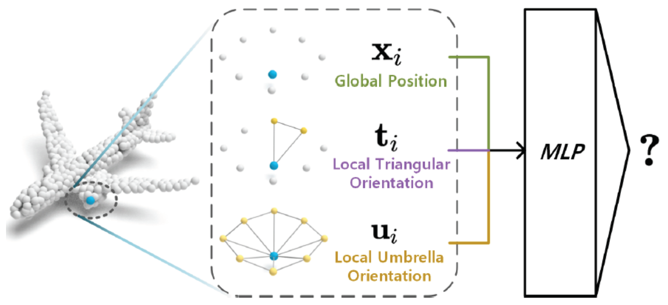


Figure 3.6: Architecture of RepSurf [159].

Hu et al. [161] employed a multi-view CNN to extract features from photogrammetry and to link it with a multi-layer perceptron (MLP) to segment the point cloud of a bridge. Lee et al. [162] added contextual features by kd-tree and K-nearest neighbors (KNN) search to PointNet [152] and deep graph convolutional network (DGCNN) architectures [148] and improved the performance of these models for the semantic segmentation of bridges. Xia et al. [163] used the local reference frame (LRF) of points and proposed a local descriptor to extract point features in order to segment the PCD of bridges. Jing et al. [164] augmented the point cloud dataset of bridges with synthetic point clouds (Figure 3.7) and proposed Bridge-Net as a 3D DL model for semantic segmentation of arch bridge point clouds. They also used RANSAC for model-to-cloud fitting and extracting the value of parameters. Yang et al. [165] also augmented the PCD of bridges with synthetic data and developed a DL model based on the super point graph (SPG) architecture [155] for semantic segmentation of bridge PCD.

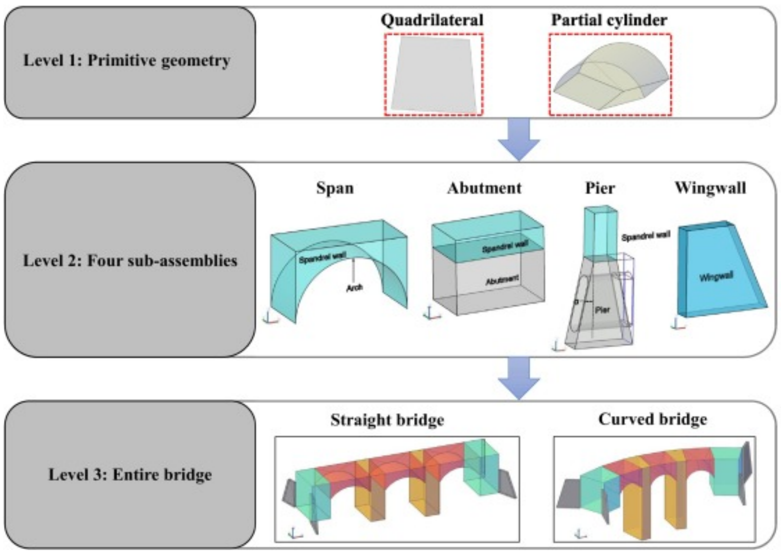


Figure 3.7: Generating synthetic point clouds using the bridge 3D model [164].

In contrast to bottom-up and top-down methods, DL models require a dataset for training. Despite the proposal of methods for generating synthetic point clouds of bridges and using them in the training phase of models [166, 164], data acquisition still seems inevitable. Furthermore, the collected point cloud samples need to be annotated to satisfy the requirements of a supervised learning schema. The annotation process is generally conducted manually, typically using the methods mentioned in Section 2.6, intensifying the manual effort required. However, DL models, after training (trained models), generally show robust performance in classifying points without the need for using conventional thresholds that often exist in top-down and bottom-up methods. Furthermore, DL models are highly adaptable and flexible and can still work even if the input samples have differences with the training dataset, i.e., they are not dependent on a set of presumptions to be satisfied. Moreover, they have a lower inference time for classifying points and can simply segment large-scale point clouds with millions of points. Most importantly, recent networks include the required preprocessing steps, such as subsampling, and can directly process raw point clouds, leading to a higher level of automation.

3.2 Bridge Model Reconstruction from PCD

Point clouds are sparse representations of real-world objects or scenes and contain a collection of discrete points in 3D space. They do not inherently convey the surface characteristics or geometry of objects. They are essentially unstructured datasets, lacking the interconnectivity information defining the relationships between points. The density of a point cloud might vary depending on the data acquisition method and the distance of the scanner; some locations might be highly dense, and others quite sparse. Furthermore, point clouds generally need a large storage space as every point in a point cloud is stored as an individual data sample.

To recognize the underlying geometry of point clouds and represent them more efficiently, they need to be summarized and reconstructed properly. In computer graphics, *surface modeling* is a method to generate surfaces from point clouds, allowing for efficient rendering, editing, and analysis of 3D objects. Tessellated Surface Representation (TSR) and Polygon/Mesh Representation (PR/MR) can be considered as surface modeling techniques. The final model of TSR-based representation or PR-/MR-based representation is a collection of connected surface elements. This type of representation using polygonal facets or polygon mesh is the most popular representation in computer graphics [167] and can be used to visualize a bridge model from its point cloud, as shown in Figure 3.8.

However, in bridge model reconstruction from PCD, the main objective is to find access to geometric details such as the dimension of elements, volume, and their spatial relationships. Polygon meshes cannot result in a volumetric body and provide geometric details about bridge elements. They also suffer from occlusions that commonly exist in point clouds. The degree of detail depends on the resolution of the polygon mesh. Increasing the mesh resolution can enhance the final model's quality, albeit at the expense of prolonging the rendering time and elevating the complexity of the representation. For instance, an essentially smooth surface might be represented unnecessarily by thousands of polygon facets [169]. To address these problems, another modeling approach in CAD,

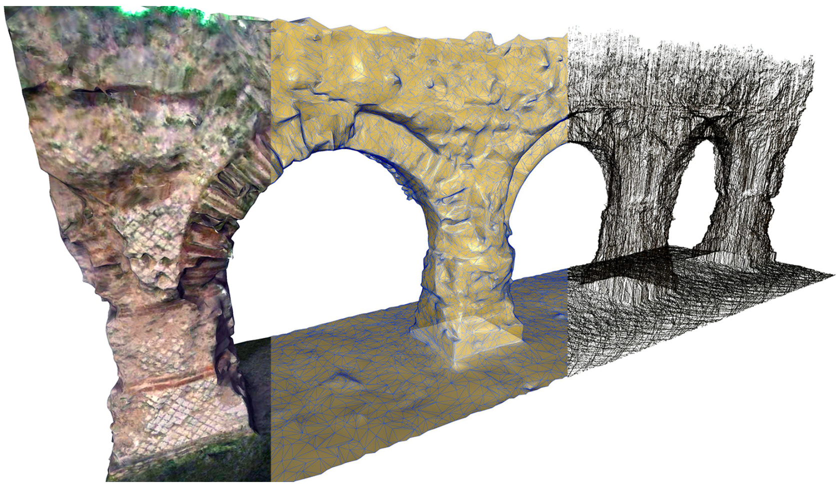


Figure 3.8: Surface representation of a bridge point cloud using meshes [168].

called *solid modeling*, is used that allows the volumetric representation and manipulation of elements through a series of additive and subtractive operations.

This section reviews solid modeling approaches that can be used principally or partially to reconstruct bridge models from PCD. Then, a summary of the methods providing the inputs for these approaches is presented, and the related works in the scope of this research are discussed.

3.2.1 Solid Modeling

Solid modeling is a technique in CAD and computer graphics that enables the creation and adjustment of 3D objects. Solid modeling approaches can be categorized into *implicit representation*, *boundary representation*, *procedural modeling*, and *parametric modeling*. Each approach has its advantages and limitations. In this section, the concept behind each approach is elaborated, and use cases are presented.

3.2.1.1 Implicit Representation

Implicit representation is a solid modeling approach to representing 3D shapes by a mathematical formulation. In computer-aided geometric modeling, implicit functions can describe a geometric space in 3D as $f(x, y, z) > 0$ [170] and a geometric surface as $f(x, y, z) = 0$ [171]. Common implicit surface definitions are plane, sphere, torus, and quadratic surfaces, as shown in Figure 3.9. The unique definition of an implicit solid model requires a set of parameters, which generally describe the origin and dimensions of the shape. The closed-form definition of implicit shapes makes them derivable and simple to evaluate. For instance, the equation of a sphere can be simply derived, and the distance of points to its surface is calculated. A mathematical descriptor is a very concise

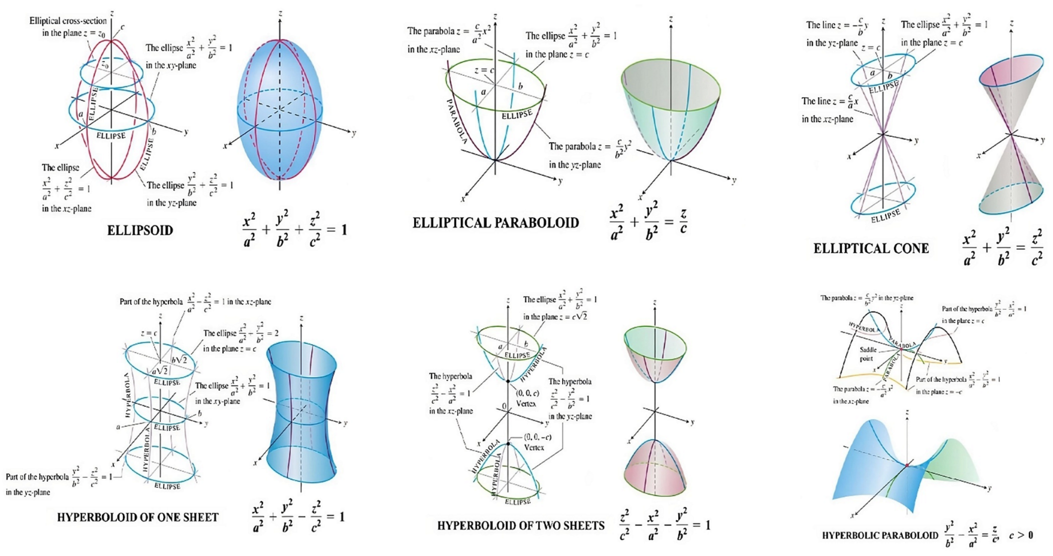

ELLIPSOID
$\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}=1$
ELLIPTICAL PARABOLOID
$\frac{x^2}{a^2}+\frac{y^2}{b^2}=\frac{z}{c}$
ELLIPTICAL CONE
$\frac{x^2}{a^2}+\frac{y^2}{b^2}=\frac{z^2}{c^2}$
HYPERBOLOID OF ONE SHEET
$\frac{x^2}{a^2}+\frac{y^2}{b^2}-\frac{z^2}{c^2}=1$
HYPERBOLOID OF TWO SHEETS
$\frac{z^2}{c^2}-\frac{x^2}{a^2}-\frac{y^2}{b^2}=1$
HYPERBOLIC PARABOLOID
$\frac{y^2}{b^2}-\frac{x^2}{a^2}=\frac{z}{c}, \; c>0$

Figure 3.9: Quadratic surfaces described by a closed-form formulation [172].

and efficient form of describing geometry, facilitating interaction with the geometry using a set of parameters. For instance, a sphere with the center (x_c, y_c, z_c) and the radius r can be simply defined using a function (mathematical descriptor) such as $(x-x_c)^2+(y-y_c)^2+(z-z_c)^2=r^2$. The defined sphere can be adjusted using its parameters (center and radius), leading to a new sphere.

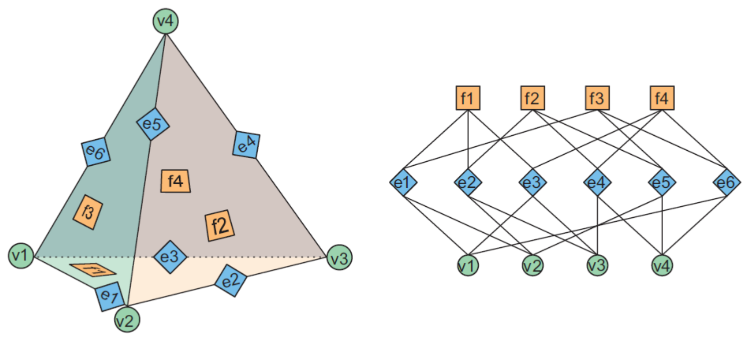
Implicit solid modeling is capable of generating volumetric bodies with a concise/abstract definition. Furthermore, it provides a direct accessibility to the value of parameters or dimensions describing the geometry. However, it has limitations in defining sharp features such as edges and vertices [173], as these features cannot be simply described in a closed-form formula. Also, implicit solid models are limited to only a number of shapes that can be mathematically expressed through a set of parameters.

Given the segmented point cloud of a bridge, implicit modeling cannot solely describe the entire scene. This is mainly due to the complex geometry of the bridge components and the limitation of implicit modeling in shapes that cannot be described simply using a set of mathematical functions. In addition, implicit modeling only provides a piece-wise solution for the geometric representation of objects. Therefore, the reconstruction might yield segments with individual functions lacking topological integration.

Even though implicit models come close to what is known as parametric models, they lack the constraint-based co-dependency between representations. It means when updating an element geometrically, its neighboring solid models cannot interact properly, as implicit modeling does not typically define the dependencies and relationships among objects. The structure of the model is thus flat and limited to the segments for which a mathematical function can be described. To increase topological integration in an implicitly modeled scene, an extension with procedural models or parametric models is needed.

3.2.1.2 Boundary Representation

Boundary representation (B-rep) is an explicit method that uses boundaries to describe a shape. In B-rep, an object is expressed by a hierarchical data structure assembling the solid by describing a network of connections between faces, edges, and vertices [174]. A vertex is defined by its (x, y, z) coordinates, an edge can be a straight or curved line connecting the vertices, and a face can be in a parametric form connecting the boundaries. This system of relationships defines the topology of the model and can be represented by a data model such as a graph, which is known as *vertex-edge-face* graph or vef graph [175], as shown in Figure 3.10.



solid	faces
1	1,2,3,4

face	edges
1	1, 2, 3
2	2, 4, 5
3	1, 5, 6
4	3, 4, 6

vertex	coordinates
1	2, 0, 0
2	0, 0, 0
3	3, 0, 0
4	1, 1, 3

edge	vertices
1	1, 2
2	2, 3
3	3, 1
4	3, 4
5	2, 4
6	1, 4

Figure 3.10: B-rep data model for representing a simple pyramid [175].

The topological information can be supplemented with geometric dimensions to describe the geometric body fully. If a geometric body has only straight edges and flat surfaces, geometric information is only required for the nodes, i.e., the coordinates of the vertices. For the curved edges, the geometric information describing their shape or curvature is also required [175].

When aiming to reconstruct an entire scene from point clouds, the B-rep method is capable of modeling more free-form objects than implicit representation. However, B-rep tends to consume more storage space to represent geometries, especially when modeling highly irregular and complex geometries. Also, the data model of B-rep needs to be extended to represent more complicated geometries containing openings or cut-outs.

3.2.1.3 Procedural Modeling

Procedural modeling is a technique that relies on creating and subsequently varying a set of base models through a sequence of rules, instructions, or algorithms. The core concept of procedural modeling is to store not only the outcome of a modeling process but also the sequence of creating objects and performing modeling operations, called the model construction history. Models created this way are called *procedural models* or *construction history* models.

Constructive Solid Geometry (CSG) can be mentioned as a pioneering procedural modeling method in which primitive shapes such as spheres, cylinders, cones, and so on are combined to create the desired model [176]. Intersection (∩), union (∪), and difference (-) are Boolean operations to create a CSG, as shown in Figure 3.11.

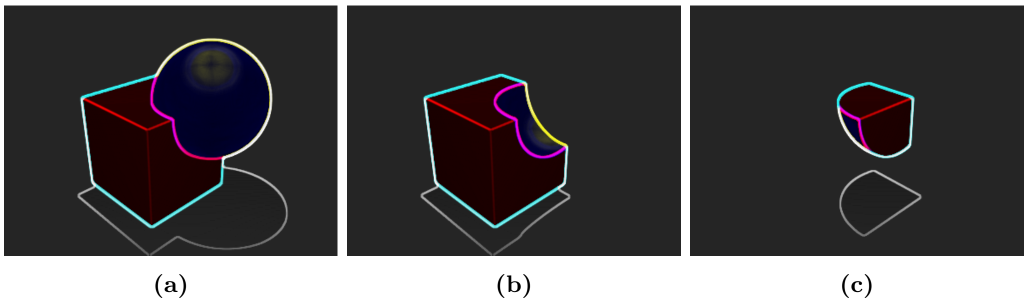


Figure 3.11: Boolean operations in CSG: (a) Union; (b) Subtract; (c) Intersect [177].

3D CAD and BIM systems have adopted the principle of Boolean operators and extended their functionality significantly by making it possible to apply them to previously modeled 3D objects [178, 179]. This offers a powerful tool for intuitively modeling complex 3D objects. In the context of BIM, the definition of subtraction solids plays an important role in the modeling of openings and penetrations [180].

Contrary to CSG, the base models of procedural models are not limited to primitive shapes. Procedural models typically use the concept of *Sweeping and Extrusion*, another solid modeling approach in some references, to create 2D sketches and transform them to a 3D geometry by Extrusion, Sweep, Loft, or Rotation CAD functionalities [181], as shown in Figure 3.12. In procedural modeling, the volumetric geometries generated by these functionalities are combined using Boolean operations to create a model with a higher level of complexity. In doing so, not only can the sequence of operations be stored efficiently, but highly complex geometries that satisfy most modeling requirements are also obtained.

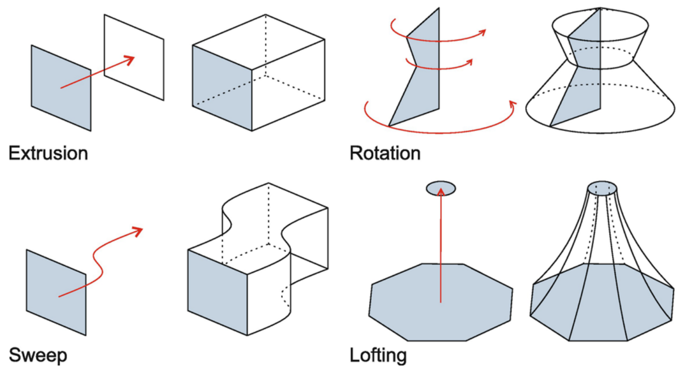


Figure 3.12: CAD functionalities to transform a 2D sketch to a 3D solid model [12].

A procedural model is modular as it is expressed through a set of assembled components. It is also auto-variant, meaning each modular component's dimensions and shape can be modified. One of the great features of procedural modeling is the capability of sharing data among various module components, leading to a smaller file size. Procedural modeling can provide a volumetric model with a link through which semantics, such as the geometric details of the modular components, are extracted. However, updating a procedural model still needs to be conducted at a component level and mostly manually. The lack of geometric constraints and parameters between the representations hinders a seamless automatic model update.

3.2.1.4 Parametric Modeling

A fundamental problem in CAD is how to make intuitive knowledge explicit such that a machine can interpret and treat it automatically [182]. Parametric modeling is a design approach that leverages known geometric information to create a dynamic model. This concept was developed in the 1990s [183] to capture design intent based on a set of features and constraints. While applied primarily in mechanical engineering, the concept has also been increasingly used to create adaptable models of infrastructure facilities [184, 48]. Two-dimensional parametric sketches form the basis of a parametric model. They are composed of geometric objects and parametric constraints. In a parametric model, particular dimensions such as positions, heights, and widths are defined using variables instead of fixed numerical values. This feature aids designers in altering a design or exploring different variants immediately, as shown in Figure 3.13. The set of parametric constraints that all major constraint solvers implement is defined as the standard geometric constraint language [185]. It comprises the dimensional constraints for distances and angles and geometric constraints to preserve the geometric shape.

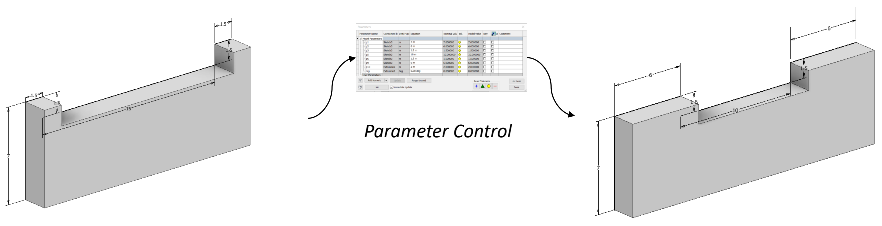


Figure 3.13: Adjusting parameters value of a parametric model.

Contrary to the aforementioned solid modeling approaches, parametric modeling maintains the resources and relations between parts/volumes in a parametric format. As a result, the geometry of a parametric model has the ability to be modified by changing the value of parameters [186]. A procedural model can become a parametric model if a set of functions expresses the relations among modular components and their dimensions. Using this set of functions, the modular components can interact with their counterparts. They also contain a set of geometric constraints that control and preserve their desired shape while being updated [48]. Geometric constraints can, for example, define that two boundary-defining lines (e.g., of different building elements) must meet at their ends and that two lines are perpendicular to or parallel to one another. Dimensional constraints, on the other hand, define only dimensional values such as length, distance, or angle.

In the scope of geometric digital twinning, a parametric model can provide an access point for updating the existing geometry based on a set of input values. As a result, it can provide a bidirectional link to import new geometric features into the model and export the existing features from the model. This considerable feature makes parametric models highly flexible and a more appropriate method for geometric digital twins that must be capable of receiving geometric updates over their service life. Nonetheless, the modeling process of parametric models is generally more time-consuming and labor-intensive than those of the other solid modeling approaches. The functional implementation of the access point in parametric models provides an extensive topological integration, allowing the attachment of semantics at various levels. The access point can receive location-based queries and return the model sub-views depending on the use cases.

3.2.2 Reverse Engineering in CAD

Reverse engineering is the process of dismantling a system or model to realize how it accomplishes a task. All reverse engineering processes consist of three basic steps: *Information Extraction*, *Modeling*, and *Review* [187, 188]. Information extraction is the process of gathering information from the desired system. Modeling is acquiring and combining data to create the geometry, and review is the testing process of the resulting model.

In CAD, reverse engineering has been a fundamental problem addressed with various techniques over the years [189, 190]. The general framework of reverse engineering in CAD has been shown in Figure 3.14. At the beginning of the process, the desired model to achieve is assumed. In parallel, the input data (such as a point cloud) is collected. The main objective of this technique is to convert the data to the desired model. Thus, the input data is directed such that it can lead to the desired model at the end of the process. Pre-processing, segmentation, feature classification, and modeling are the middle steps to generate the model from the scanned data [191]. Each part is designed to produce the anticipated model; for instance, the classes required for semantic segmentation are determined based on the final model.

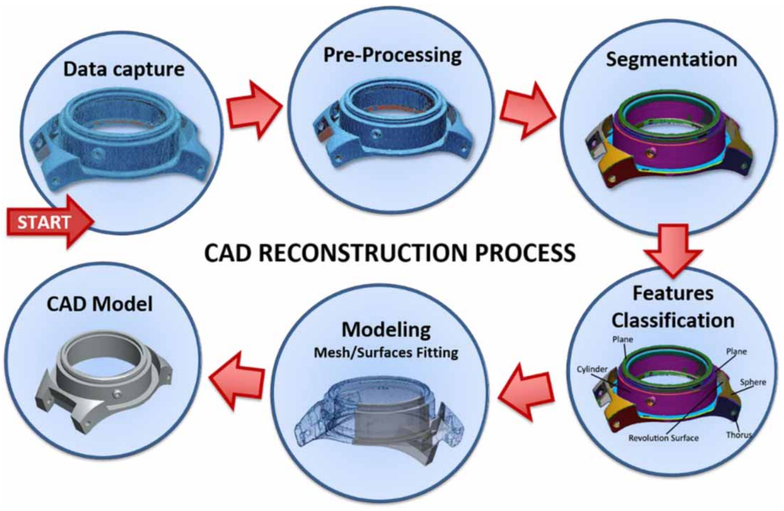


Figure 3.14: Reconstruction of a watch-case based on reverse engineering [191].

Reverse engineering can facilitate the model creation process from scanned data through parametric modeling. Depending on the model type the scan data represents, the parametric model of the object can be created. Due to the parametric design of the model, it can be compared (reviewed) with the scanned data and be further altered to reach a higher level of similarity. Recently, this CAD approach has also been of interest to leverage prior knowledge about the topological and existing rules in PCD to model the geometry of objects [192, 193].

3.2.3 Geometry Reconstruction

Various methods have been proposed to model the geometry of three-dimensional bodies from PCD automatically or semi-automatically. The proposed approaches generally provide the inputs for solid modeling approaches to represent a geometry with a desired level of abstraction or details. Leveraging the closed-form description of primitive shapes and providing an objective function to evaluate the closeness of primitives to points, various techniques have been proposed to address the model-to-cloud fitting problem. On top of them, the RANdom SAmple Consensus (RANSAC) algorithm [124], Hough transform [194], and least squared optimization algorithms [195] can be mentioned. Most recently, deep learning models have also been capable of using the objective function of primitive shapes (Figure 3.15) to automate the simultaneous semantic segmentation and geometric modeling of primitive shapes [160].

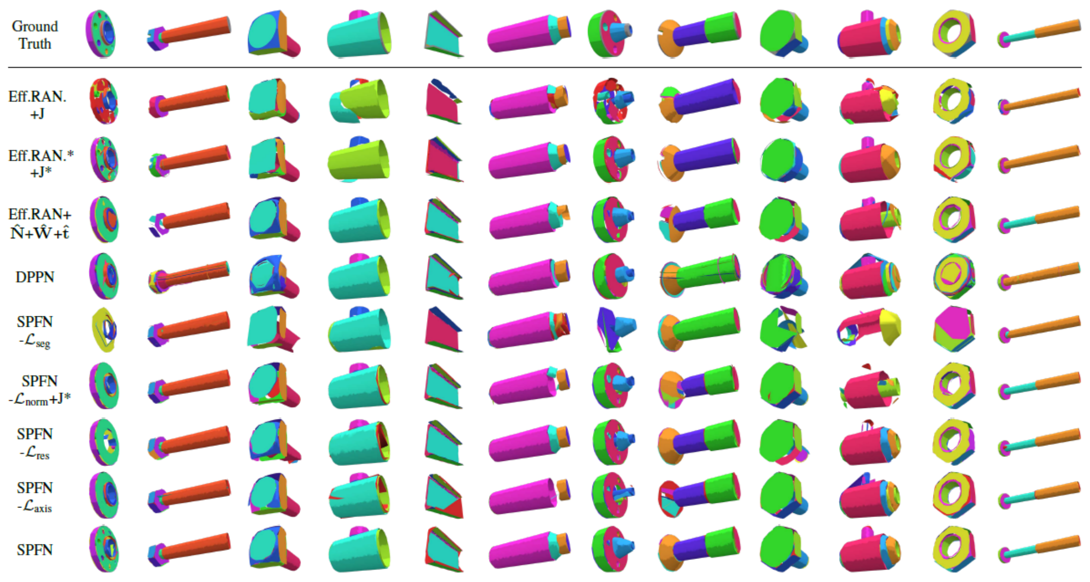

Ground Truth
Eff.RAN. +J
Eff.RAN.* +J*
Eff.RAN+ $\hat{N}$+$\hat{W}$+$\hat{t}$
DPPN
SPFN $-\mathcal{L}_{seg}$
SPFN $-\mathcal{L}_{norm}$+J*
SPFN $-\mathcal{L}_{res}$
SPFN $-\mathcal{L}_{axis}$
SPFN

Figure 3.15: Primitive fitting results using deep learning models [160].

B-rep methods have also been used to construct low-semantic and generic models such as meshes/patches from point clouds to address the emerging challenges of modeling more complex shapes whose description by a closed-form formula is cumbersome. To reduce the unwanted complexity of meshes in modeling and storing the geometry, bounding hulls such as convex hull [196], α-shape [197], x-hull [198], concave hull [199], crust

[200], etc. have been introduced as well. These methods generally result in the explicit representation of the boundary points. They can illustrate the geometry of complicated shapes solely or in combination with CAD functionalities such as extrude, loft, rotate, and sweep. They, however, cannot simply/directly provide meaningful information about the required parameters to create a volumetric model.

Lu and Brilakis [201] created the geometric DT of bridges from point clouds using a 2D ConcaveHull α-shape method [199] and generated 3D shapes using Industry Foundation Classes (IFC), as shown in Figure 3.16. Zhang et al. [202] detected the planar patches from noisy point clouds and determined the boundaries of each patch by the α-shaped algorithm. Wang et al. [203] employed the M-estimator SAmple Consensus (MSAC) algorithm to detect the planar faces and extracted the value of parameters from regular and irregular shapes through a line detection algorithm. Yang et al. [204] employed the principal component analysis (PCA) algorithm to detect the alignment of elements and extracted the value of parameters using the RANSAC algorithm [119].

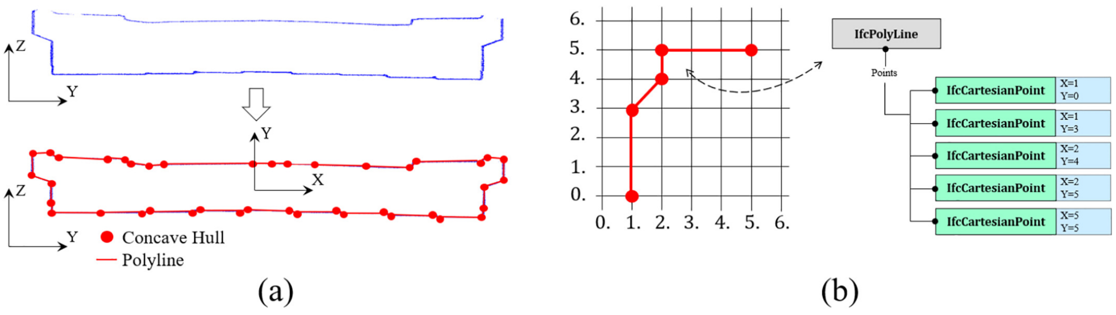

Z
Y
Y
X
Z
Y
● Concave Hull
— Polyline
(a)
6.
5.
4.
3.
2.
1.
0.
0. 1. 2. 3. 4. 5. 6.
IfcPolyLine
Points
IfcCartesianPoint X=1 Y=0
IfcCartesianPoint X=1 Y=3
IfcCartesianPoint X=2 Y=4
IfcCartesianPoint X=2 Y=5
IfcCartesianPoint X=5 Y=5
(b)

Figure 3.16: Geoemtric modeling of a bridge deck: (a) concave hulls; (b) IfcPolyline object [201].

Dimitrov et al. [115] proposed an approach for successively fitting uniform B-Spline curves to the two-dimensional cross-section of point clouds. Kwon et al. [205] described a heuristic method for extracting the value of parameters from primitive shapes such as cuboids and cylinders. Justo et al. [206] generated the IFC model of truss bridges using bounding boxes of instance-segmented point clouds and collision of elements. Jing et al. [164] employed RANSAC for extracting the geometric features from segmented bridge point clouds and created the 3D model of bridge elements. Valero et al. [207] detected the planer surfaces in the point clouds and determined the value of parameters by measuring the distance between planes. Oesau et al. [169] proposed a rough feature preserving multi-scale line fitting and a graph-cut formulation to reconstruct a building point cloud into a mesh-based model. Rabbani [195] proposed a method based on least-squared optimization to model a piping system from its point cloud.

Patil et al. [208] suggested an area-based adaptive hough transform to estimate single and multiple cylinder orientations and reconstructed piping networks by finding the connection relationships between pipes. Walsh et al. [209] segmented the point cloud of structural elements using features such as normal vectors, curvature, and connectivity of points and extracted the value of parameters from primitive shapes using a least-squares optimization algorithm. Laefer and Truong-Hong [210] proposed a kernel density estimation (KDE) algorithm to detect the density signal of steel profiles and match them with the standard sections in a catalog.

Yan and Hajjar [132] employed the RANSAC algorithm to detect the plane surfaces of steel profiles and model the super-structure components of bridges. Kim et al. [193] presented an approach based on reverse engineering for segmenting pipe point clouds through deep learning models and employed a 3D matching system to reconstruct 3D plant models, as shown in Figure 3.17. Li et al. [160] described a deep learning model to segment and estimate the parameter values of primitive shapes from point clouds. Barazzetti [211] proposed an approach for the parametric as-built model generation of complex shapes from point clouds using NURBS curves and surfaces.

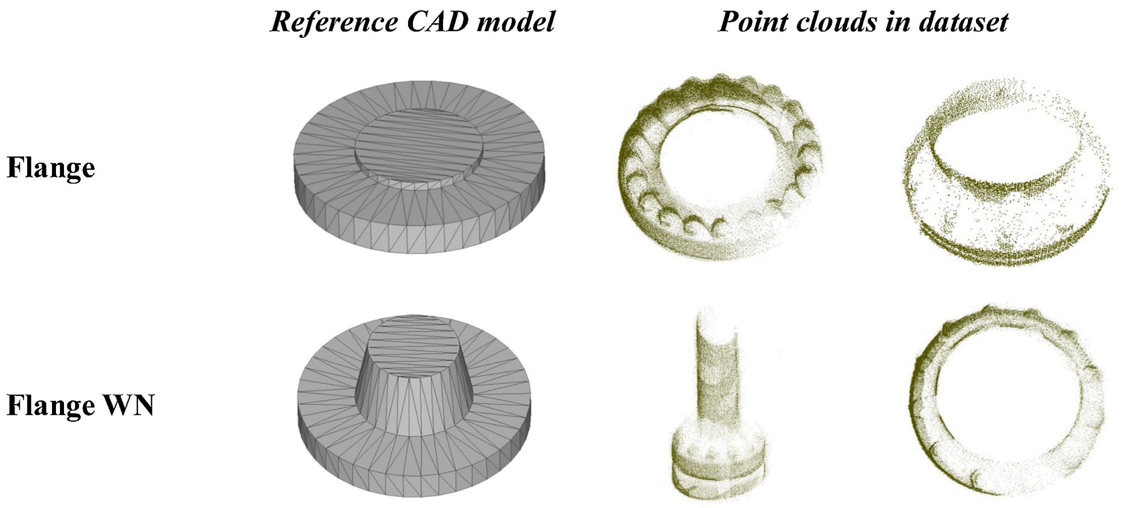


Figure 3.17: Comparison of the modeled element with segmented point clouds for flange and flange WN [193].

3.3 Research Gaps

Despite the impressive progress in the geometric digital twinning process of bridges, several research gaps still need to be addressed. Some of the limitations and the parts requiring further investigation are as follows:

> *Gap 1:* The existing DL models capable of processing points have not been expressed specifically for bridges. These models are often highly generic and require further development for the efficient semantic segmentation of bridge point clouds.
>
> *Gap 2:* The existing models can be equipped with cutting-edge AI modules/layers such as non-local attention, self-attention, and contextual attention for efficient encoding of points.
>
> *Gap 3:* Most existing models only encode local neighborhoods and cannot provide the existing spatial relationships between points located far from each other.
>
> *Gap 4:* Despite the existence of mechanisms to fit primitive shapes into point clouds, it is not crystal clear how non-primitive shapes can be modeled automatically. Most bridge elements, such as the bridge deck, abutment, and parapet, are not primitive shapes.
>
> *Gap 5:* The proposed algorithms mostly follow a bottom-up approach. Hence, they require the setting of many problem-specific thresholds, and their performance is highly affected by occlusion, which is a common problem in practice.
>
> *Gap 6:* The final 3D model in similar works is not parametric, i.e., it cannot update its shape as the value of parameters changes. Hence, the model loses its link (access point) to the actual asset for any further geometric update, and such updates should be applied at the component level.
>
> *Gap 7:* It has not been adequately investigated how the elements are assembled into a coherent model after the initial element-wise model fitting. This aspect is even more relevant when the components are parametric, and the final model needs to preserve its parametric consistency.

This thesis addresses the aforementioned research gaps by automating the semantic segmentation and parametric modeling of bridges from their point clouds. The research gaps 1-3 correspond to the semantic segmentation of bridge point clouds and are addressed by proposing a novel DL architecture. This model benefits from cutting-edge

AI tools and can improve the segmentation accuracy. Research gaps 4-7 are concerned with the model reconstruction of bridges. They are resolved through a novel optimization method that can generate and assemble the parametric model of bridge components from their segmented point clouds. The proposed research framework is explained in more detail in the following chapters and tested in automated geometric digital twinning of actual bridges.

4 Automated Semantic Segmentation

Semantic segmentation is the essential first step in the proposed framework for the automated 3D geometric modeling of bridges from PCD. This process differentiates the point cloud of a bridge into separate point clouds of the bridge elements, thereby paving the way for further extraction of geometric and semantic features. Despite the existence of tools and software programs for manual semantic segmentation of bridge point clouds, conducting this step is yet laborious, error-prone, and costly.

To address this issue, this section proposes a novel deep learning model, coined Multiscale Spatial Feature Descriptor (MSFD-Net), to automate the semantic segmentation process of bridge point clouds. The architecture of this model can be seen in Figure 4.1. It consists of various modules to extract local as well as global features and provides a mechanism for describing the pair-wise relationships between points. The local features are calculated in the Point Feature Descriptor (PFD) module and combined with the global features using a sum function. MSFD-Net expresses point features in various scales to leverage the low-level as well as high-level features in the label prediction process.

Similar to other 3D DL architecture, the number of encoding blocks can vary, leading to a larger/deeper architecture. However, we define MSFD-Net with four encoding/decoding as well as four scales for generating point features. Each encoding block of MSFD-Net is followed by a random subsampling (RS) module to reduce the processing load of the network. Thanks to this subsampling strategy in the subsequent layers of the network and a U-shaped autoencoder, MSFD-Net is capable of processing large-scale bridge point clouds.

The bottleneck of this architecture contains Non-Local Attention (NLA) modules to encode the spatial relationships between points located far from each other. This chapter describes the architecture of MSFD-Net. It breaks down the model into its constitute modules and explains the concepts behind them in more detail. This model is then used in Section 7.2 for the semantic segmentation of single-span RC bridge point clouds.

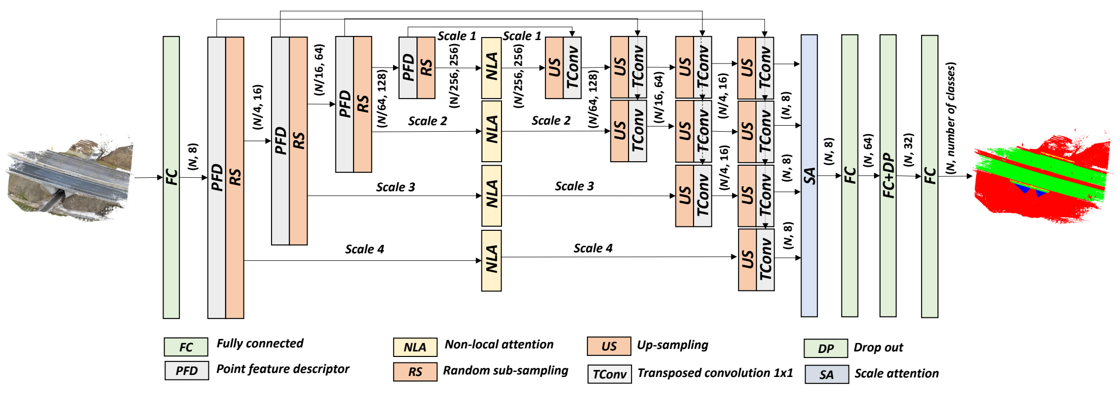


Figure 4.1: Architecture of MSFD-Net.

4.1 MSFD-Net architecture

MSFD-Net has been designed to capture the local and global features as well as the spatial dependencies among points and model the existing patterns in various scales. It requires the coordinates of points (x, y, z) and their corresponding normal vectors (n_x, n_y, n_z) as input to determine the semantic label of points in a point cloud. Other features, such as RGB color codes, density, and even learned latent features, can also be added and processed by this network. As shown in Figure 4.1, the architecture of MSFD-Net is basically an auto-encoder (U-Net) expressed in several scales. The model's encoder includes a Point Feature Descriptor (PFD) module followed by random sub-sampling in sequential layers. Leveraging the sub-sampling and aggregation mechanisms, MSFD-Net can process large-scale point clouds.

The PFD module receives point features to generate a geometric representation of the local neighborhoods. It also aggregates the generated local features and global features and increases the receptive field of points. The bottleneck consists of Non-Local Attention (NLA) modules providing spatial attention (pairwise dependencies) about the point features in the latent space. The decoder has also been expressed in multi-scales to fuse the high-level feature map of points with the basic features generated by the initial and intermediate layers. The first scale of this model is a plain decoder collecting features from all the layers of the network while the next scales ignore some of the intermediate blocks to provide a better perception of the initial layers. The decoded spatial features are attentively fused in the end to generate a feature map from all the scales. These modules are further described in the following sections.

4.2 Point Feature Descriptor (PFD)

As described, semantic segmentation of point clouds is a classification process at the point level. DL architectures generate a set of features for each point and employ it to predict class labels. While a single centroid point (i.e., a typical point in a point cloud to be classified) might not offer extensive details about its underlying patterns, examining it in conjunction with its neighboring points can yield enriched information about these patterns. Local features are the features described at a point level considering the neighboring points of each centroid point. In combination with global features, they provide comprehensive information that aids the network in classifying points.

PFD aims to extract and learn the geometric features of points. It also combines local and global features to provide a more comprehensive description of the scene. The framework of the PFD has been inspired by the local feature aggregation module proposed in RandLA-Net [158]. However, its feature encoder is different and consists of two blocks named Local Spherical Representation (LSpR) and Local Surface Representation (LSuR). These modules are employed in MSFD-Net to provide a more comprehensive expression of the local neighborhoods. The details of these blocks are elaborated on in the next sections, and then, a general overview of the feature aggregation module is provided.

4.2.1 Local Spherical Representation (LSpR)

Point-based DL models often only adopt Cartesian coordinates for describing local neighborhoods, while the relative position of points is highly sensitive to the possible transformations and normalization in this coordinate system. On the contrary, the spherical coordinate system can demonstrate the position of a point by two angles limited in the range of $[0, 2\pi]$ and a radius/distance, leading to a less sensitive representation. Figure 4.2a shows the local neighborhood of the centroid point p_i and its K neighbors $\{p_i^1, p_i^2, ..., p_i^K\}$ obtained by K-nearest neighbors (KNN) algorithm. The relative position of p_i with respect to its neighbors can be expressed in the spherical coordinate system $(\rho_i^k, \phi_i^k, \theta_i^k)$ as below:

$$\rho_i^k = \sqrt{x_i^{k^2} + y_i^{k^2} + z_i^{k^2}} \tag{4.1}$$

$$\phi_i^k = arctan(\frac{y_i^k}{x_i^k}) \tag{4.2}$$

$$\theta_i^k = arctan(\frac{z_i^k}{\sqrt{x_i^{k2} + y_i^{k2}}}), \tag{4.3}$$

where (x_i^k, y_i^k, z_i^k) are the relative coordinates of p_i^k in the Cartesian coordinate system with center p_i.

To prevent information loss, the normalized relative position of neighboring points with respect to the centroid point is also encoded. This leads to three more features $(x_i^{k'}, y_i^{k'}, z_i^{k'})$ describing the local neighborhood using unit vectors, where $x_i^{k'} = x_i^k/\rho_i^k$, $y_i^{k'} = y_i^k/\rho_i^k$, and $z_i^{k'} = z_i^k/\rho_i^k$. Finally, all these six features are concatenated, leading to the feature set $f_{LSpR} = \rho_i^k \oplus \phi_i^k \oplus \theta_i^k \oplus x_i^{k'} \oplus y_i^{k'} \oplus z_i^{k'}$, where $\oplus$ is the concatenation operation.

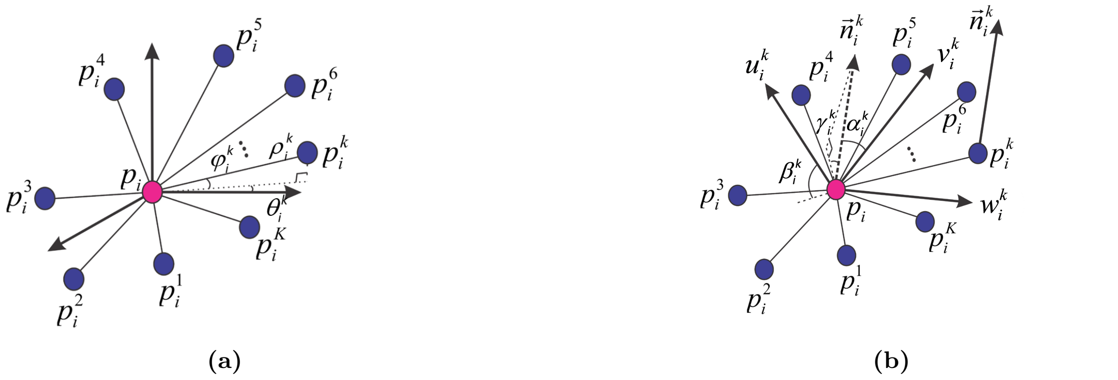


Figure 4.2: Local representation of a neighborhood: (a) local spherical representation (LSpR); (b) local surface representation (LSuR).

4.2.2 Local Surface Representation (LSuR)

The generated features by LSpR describe the relative position of points in a local neighborhood. They can be a basis for extracting further features from the local neighborhoods. However, they might yet lack adequate information about the representative pattern by points. To handle this problem, a local Darboux frame can be defined, and the underlying surface on which the points are positioned is approximated [212]. A Darboux frame is a dynamic/moving frame constructed on a surface. It is the analog of the Frenet–Serret frame and can represent the curvature, normal curvature, and relative torsion of the surface.

Figure 4.2b illustrates the query point p_i with the normal vector $\vec{n}_i$ as well as the neighboring points $\{p_i^1, p_i^2, ..., p_i^K\}$ and normals $\{\vec{n}_i^1, \vec{n}_i^2, ..., \vec{n}_i^K\}$. A Darboux frame with the axes $(\vec{u}_i^k, \vec{v}_i^k, \vec{w}_i^k)$ can be defined for each pair of (p_i, p_i^k) as below [212]:

$$\vec{u}_i^k = \vec{n}_i, \quad \vec{v}_i^k = \vec{u}_i^k \times \frac{p_i^k - p_i}{||p_i^k - p_i||_2}, \quad \vec{w}_i^k = \vec{u}_i^k \times \vec{v}_i^k, \tag{4.4}$$

where $||.||_2$ is the L2 norm of the vectors connecting the centroid point to the neighboring points.

Based on the defined Darboux frame, the dependencies between each pair (n_i, n_i^k) can be defined by three angles $f_{LSuR} = (\alpha_i^k,\ \beta_i^k,\ \gamma_i^k)$ as follows [212]:

$$\vec{\alpha}_i^k = \vec{v}_i^k.n_i^k, \quad \vec{\beta}_i^k = \vec{u}_i^k.\frac{p_i^k - p_i}{||p_i^k - p_i||_2}, \quad \vec{\gamma}_i^k = arctan(\frac{\vec{w}_i^k.\vec{n}_i^k}{\vec{u}_i^k.\vec{n}_i^k}), \tag{4.5}$$

The LSuR module is dependent on the direction of the normal vectors. Changing the direction of the normal vectors leads to different values for the variables α_i^k, β_i^k, and γ_i^k. This causes instability in the network performance in scenarios where the normal vectors have an opposite direction to those of training samples. To address this issue, we augment the local neighborhoods with respect to the normal vectors by randomly changing the direction of normals in each neighborhood. This can be achieved by multiplying normal vectors randomly by -1, as shown in Figure 4.3. In doing so, the model becomes invariant to the direction of normal vectors as it has learned neighborhoods with normal vectors in both directions.

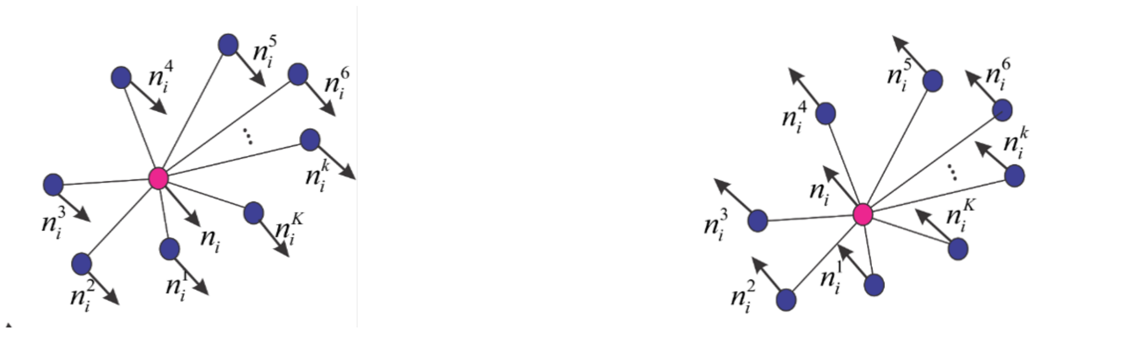


Figure 4.3: Augmentation of normal vectors on local neighborhoods: (a) original; (b) augmented.

4.2.3 Feature Aggregation

As shown in Figure 4.4, the PFD module receives a point cloud containing N points, each described by d features, i.e. $\mathcal{F} = \{f_1, ..., f_i, ..., f_N\}$, where $\mathcal{F} \in \mathbb{R}^{N \times d}$. This feature set is passed through a shared MLP ($\mathcal{M}$) to provide a global description of the input features ($\mathcal{M}(f_i)$). In parallel, the neighboring points are gathered by a KNN search algorithm, and the global features are distributed over the local neighborhoods (f_i^k). The neighboring points and normals are also sent to the LSpR and LSuR blocks to generate the local features. These features are concatenated, and a shared MLP is applied as below:

$$r_i^k = \mathcal{M}(f_{LSpR} \oplus f_{LSuR}), \tag{4.6}$$

The feature set r_i^k has been mostly defined based on rotation angles that, in turn, aid the network to learn local features and achieve more robust performance in practice. The local features r_i^k and the global features f_i^k are concatenated eventually ($f_i^k \oplus r_i^k$) to generate the new set $\hat{F}_i = \{\hat{f}_i^1, ..., \hat{f}_i^k, ..., \hat{f}_N^K\}$ describing the spatial features of points in each neighborhood. To summarize the features of the neighborhoods, symmetric functions such as mean and max can be applied. However, to emphasize the more important features, the weighted summation of features is calculated while the weights are also learnable parameters for the network. This process, which is called attentive pooling, can be described by [158]:

$$\tilde{f}_i = \sum_{k=1}^{K} (\hat{f}_i^k . g(\hat{f}_i^k, W)), \tag{4.7}$$

where $g()$ consists of a 1×1 convolution layer followed by a softmax to generate attention scores, and W is the learnable weights of the shared MLP. Contrary to RandLA-Net, which generates the weights W through a fully connected layer, MSFD-Net uses a convolution layer. This simple step can eliminate the need to change the shape of tensors and speed up the performance of the network to a large extent.

The PFD module leads to a set of features that encapsulates points' geometric and semantic features. To collect residuals and preserve the lower-semantic information incoming to the module, a skip connection is used, and two sets of features are stacked in each layer. As a result, the receptive field of points increases, and the generated features can be propagated over the neighborhoods.

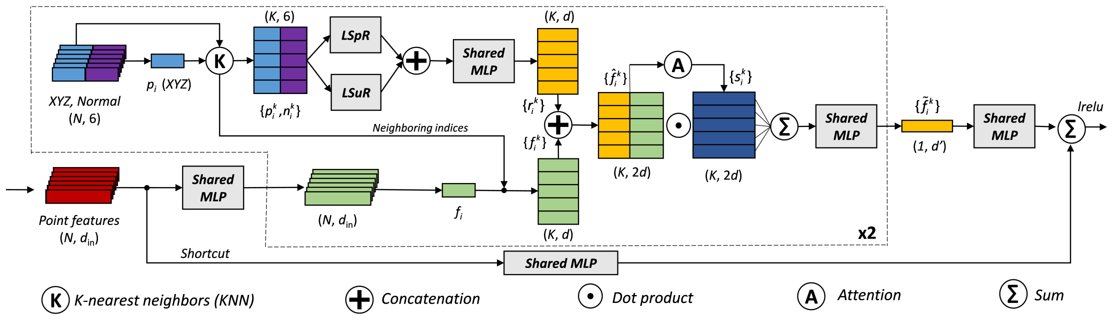


Figure 4.4: Point Feature Descriptor (PFD) module.

4.3 Spatial Attention

The PFD module presents every point based on a set of local and global features collected from its local neighborhood. It also stacks the features and propagates them to the surrounding neighborhoods to increase the receptive fields. To preserve the computation efficiency of the model, the number of stacks is generally limited and cannot be increased. This limitation gradually reduces the impact radius of points, and as a result, the pairwise dependencies (global awareness) cannot be expressed completely for the points placed at a distance.

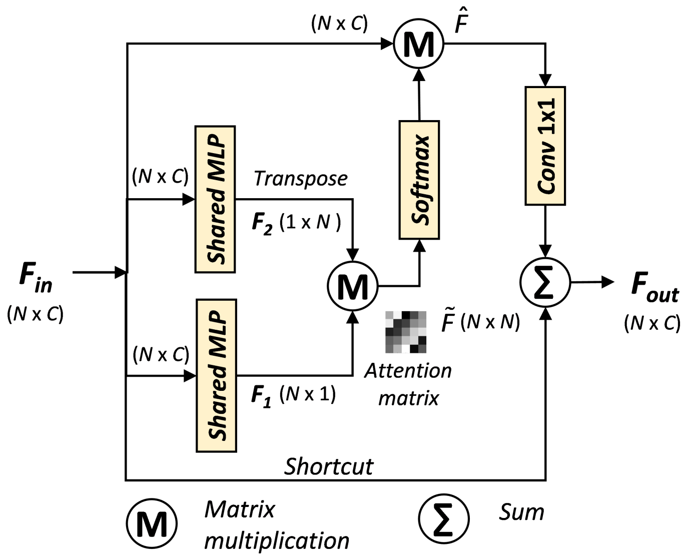


Figure 4.5: Non-local attention layer.

Non-Local Attention (NLA) is a module first proposed to encode the spatial dependencies between pixels in images [75]. This block has also been recently used in the spatial encoding of point clouds [213]. Non-local networks generally consist of three 1×1 convolution layers to provide different representations of the input in the latent space of the problem. As shown in Figure 4.5, the non-local attention module of MSFD-Net receives the input feature map $F_{in} \in \mathbb{R}^{N\times C}$ and generates the output feature map $F_{out} \in \mathbb{R}^{N\times C}$ that not only contains the initial inputs but also the spatial dependencies.

This NLA module only generates two representations F_1 and F_2 through two shared MLPs containing a 1×1 convolution layer followed by batch normalization and a Rectified Linear Activation (ReLU) function. These convolution layers generate a single feature for each point representing its spatial position in the latent space, i.e., $F_1, F_2 \in \mathbb{R}^{N\times 1}$. The batch normalization layer provides more stability, and the ReLU activation function adds non-linearity and limits the weights to positive values. The first feature map F_1 is multiplied by the transposed of the second feature map F_2 and the attention matrix $\tilde{F} = F_1 \otimes {F_2}^T$ (where $\tilde{F} \in \mathbb{R}^{N\times N}$) is obtained that encapsulates the pairwise relationship between points. This matrix is passed through a softmax function to be normalized and further multiplied element wisely by the input feature map F_{in} leading to the feature map $\hat{F} = Softmax(\tilde{F}) \odot F_{in}$. To tune the impact of $\hat{F}$ on the input feature map F_{in}, it is passed through a 1×1 convolution layer, and the resulting feature map is summed with F_{in} to generate the output feature map F_{out}. This implementation of the non-local attention module can provide more stability as the attention matrix $\tilde{F}$ is directly multiplied by the input feature map F_{in}. It also provides a more efficient mechanism to alleviate or augment the pair-wise relationships through a convolution layer before the final summation.

4.4 Scale Attention

The decoder of MSFD-Net consists of multiple scales, and the number of scales changes with respect to the number of encoding layers. These scales provide various feature maps that contain not only the high-level features but also the low-level features [214, 213]. As shown in Figure 4.1, the first scale of MSFD-Net collects the extracted features from all the layers, while the next scales ignore some of the intermediate PFD blocks to reduce the level of details. To increase the global awareness of points, the extracted features from all the scales are passed through the NLA block. The resulting feature map is concatenated with the extracted features from the corresponding encoder layer through a skip link. These features are up-sampled by nearest interpolation and passed through a 1×1 transposed convolution layer. To emphasize the more important scales in the learning process of MSFD-Net, the weighted summation of the feature maps is calculated while the weights are also learnable parameters, as shown in Figure 4.6.

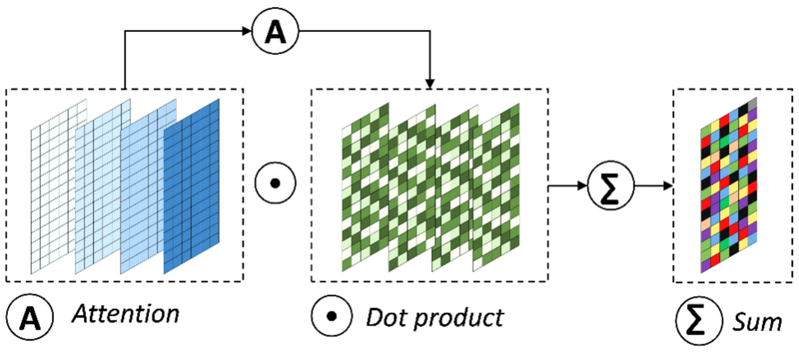


Figure 4.6: Multiscale network fusion using scale attention.

Given the feature maps $F = \{F_1, ..., F_i, ...F_S\}, F_i \in \mathbb{R}^{N\times d}$, where S is the total number of scales, N is the number of points, and d is the number of features in the last layer, the weighted summation of the feature maps can be calculated as follows:

$$\tilde{F} = \sum_{i=1}^{S}(\hat{F}_i.g(F_i, W)), \tag{4.8}$$

where $\tilde{F}$ is the final feature map, $g()$ is a function consisting of a 1×1 convolution layer followed by a softmax to generate scale attention scores, and W is the learnable weights of the shared MLP.

MSFD-Net is able to process the raw bridge point clouds and generate the segmented point cloud of bridge elements. The next question to be answered is how to generate the parametric model of each bridge element from the point cloud segments and assemble them to achieve a coherent model for the entire bridge. Before answering this question, the segmented point clouds need to be clustered/segmented, and the required 3D model for the reverse engineering approach is generated. The main objective in the two following chapters is to compare this dummy 3D model with the segmented point clouds and make it closer in shape to the scanned data by defining and solving local and global optimization problems.

5 From Semantic Segmentation to Parametric Modeling

Semantic segmentation of bridge point clouds (Module 01) yields a segmented representation of bridge components. As mentioned, it simplifies the geometric digital twinning problem for the downstream algorithms to a large extent. However, it still needs to be followed by other algorithms to provide an enriched representation of points that the second module can use properly.

Within the segmented point cloud of bridges, there might still be multiple point cloud instances. For example, two abutment instances generally exist in the point cloud of RC bridges. To enable parametric modeling using the proposed method, the segmented point cloud of classes containing more than one instance needs to be further segmented/clustered. The reverse engineering approach also needs the desired output bridge model before starting the model-fitting process. Based on this model, Parametric Prototype Model (PPM)s are designed in Chapter 6 to derive the value of parameters from segmented point clouds. PPMs can be applied to the boundary points of faces or cross-sections. Thus, these points should be detected efficiently prior to applying them.

This chapter closes these gaps by proposing and developing algorithms/methods that prepare the segmented point clouds for applying PPMs. The segmented point clouds resulting from the first major module are clustered through an unsupervised clustering algorithm. The point clusters are then de-noised, ensuring no point stands far from the target cluster. The boundary points of faces are detected through another clustering algorithm. Finally, the 3D model required for the reverse engineering approach is designed.

5.1 Clustering & De-Noising

Multiple instances exist in the segmented point cloud of classes, such as railings and abutments. To enable piece-wise model-fitting, the point cloud of these classes needs to be segmented/clustered[1]. To decrease the value of error resulting from downstream algorithms, the detected clusters can be de-noised as well. Density Based Spatial Clustering of Applications with Noise (DBSCAN) [215] is an automatic clustering algorithm proposed for discovering clusters in large spatial databases. This algorithm starts from a random point and expands the region based on the local density of data points. DBSCAN can be used to cluster and refine points in bridges [163]. However, setting a threshold value for density in bridges is challenging, especially in bridge point clouds with different resolutions. Also, it is computationally expensive and slow to process large datasets, which is common in bridges. To address these issues, a modified version of DBSCAN is proposed to cluster and de-noise segmented point clouds of bridges. As shown in Algorithm 1, this clustering method starts from a random query point and expands the region based on the connectivity of points. To reduce the complexity order of DBSCAN from $O(n2)$ to $O(k\ log(n))$, kd-tree is used as a data structure, and the neighboring points are obtained by KNN search. Any neighbor of the query point located within a predefined distance (radius) is added to the cluster of the query point, and its neighbors are also added to the list of the query point neighbors. This process is repeated for any neighboring point in the list and continues until all the points are evaluated and assigned to a cluster.

Algorithm 1 Clustering & de-noising algorithm

Input pc: point cloud; n: number of clusters (1 for the de-noising task); r: radius; k: number of neighbors; $label$: points label, initially undefined; KNN: K-nearest neighbors search; $Dist$: function to calculate Manhattan distance

```
1:  foreach $p \in pc$ do
2:      if $label(p)$ undefined then
3:          next cluster label $\leftarrow c$
4:          $label(p) \leftarrow c$
5:          Neighbors $N \leftarrow KNN(K, pc)$
6:          Neighbors of the query point $Q \leftarrow N/\{p\}$
7:          foreach $q \in Q$ do
8:              if $label(q)$ undefined then
9:                  Distance $d \leftarrow Dist(q,p)$
10:                 if $d < r$ then
11:                     $label(q) \leftarrow c$
12:                     Neighbors of the neighboring point $S \leftarrow N/\{q\}$
13:                     $Q \leftarrow S \cup Q$
14: return $label$
```

[1]This process is called instance segmentation or clustering.

Similarly to DBSCAN, this clustering method might result in many clusters in each of which the connectivity conditions have been satisfied. This algorithm is used in two applications: clustering and de-noising. In the clustering task, the largest n clusters are selected as the smaller clusters are more likely to represent noise clusters. In the de-noising task, the first largest cluster is only extracted as the points in this cluster satisfy the connectivity conditions and are far from the points belonging to other clusters.

5.2 Boundary Points Detection

A point cloud represents the external surfaces of objects in a scene. It also implicitly contains semantic and geometric information about the objects. Depending on the use case, a point cloud can be abstracted, simplified, and purposefully represented with a lower number of points. In a model-fitting process, boundary points mostly contain the geometric information of elements. Hence, the detection of these points seems necessary for fitting PPMs.

Boundary points generally have different features than interior points. Mean shift is one of those features proposed for detecting boundary points [216]. This point-level feature is expressed as each point's distance to its neighboring points' mean point. In general, boundary points show a higher shift value toward their mean point as they cannot find neighboring points all around their vicinity. To detect these points, a threshold has been defined in [216], which is based on the distance of the query point to its nearest neighbor. However, setting the value of this threshold is difficult, especially in point clouds with different resolutions.

To address this problem, a Fuzzy C-Means (FCM) algorithm is employed to automate the detection process of boundary points. FCM is an unsupervised clustering algorithm and an extension of the K-means algorithm in which the membership degree of data samples to clusters is expressed by fuzzy logic [217]. Considering the value of the mean shift, points can be divided into two clusters with sharp features (boundary points) and points with soft features (interior points). To detect boundary points, the nearest neighbors of each point are obtained by applying KD-tree and KNN search, and the value of the mean shift is computed. This feature is then passed through an FCM with two clusters. Since the value of the mean shift is higher for boundary points, the resulting cluster with the higher mean value is selected as boundary points. As a result, the proposed threshold can be eliminated, and the required points to fit PPMs are detected automatically, as shown in Figure 5.1.

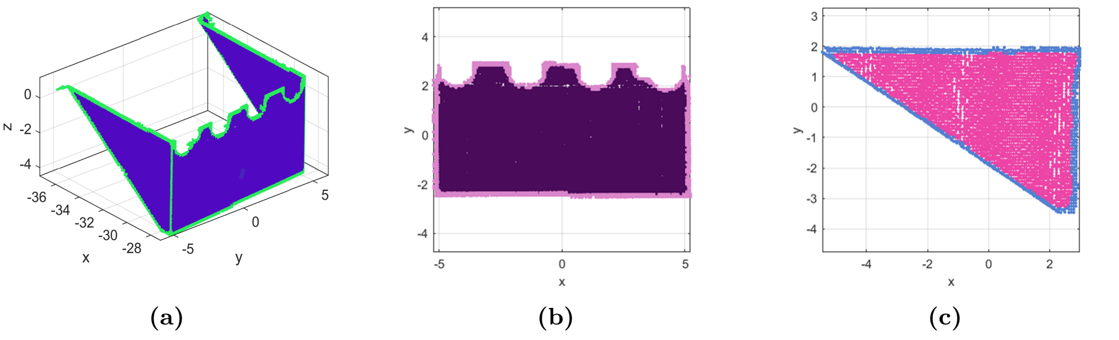
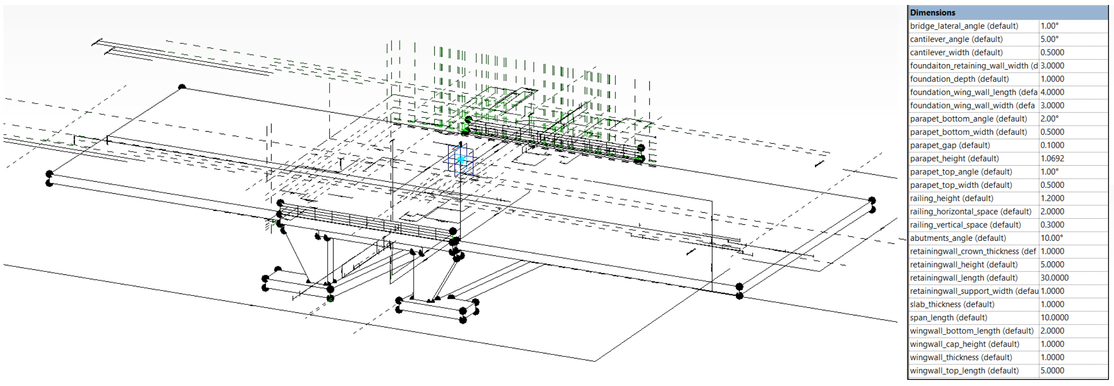


Figure 5.1: Boundary points detection by FCM clustering in 3D/2D: (a) an abutment; (b) a retaining wall; (c) a wing wall.

5.3 Model Generation

To deduct the design features of modeling the entire bridge, its 3D parametric model can be created based on a set of parameters. End users can define these parameters following a LoD satisfying the anticipated applications of the model. This user-dependent definition of the model is very close to the definition of a bridge DT as it is also created based on a set of desired use cases and requirements. Figure 5.2 demonstrates the 3D model of a single-span RC bridge created through a set of parameters to meet a desired LoD.



Dimensions	
bridge_lateral_angle (default)	1.00°
cantilever_angle (default)	5.00°
cantilever_width (default)	0.5000
foundaiton_retaining_wall_width (d	3.0000
foundation_depth (default)	1.0000
foundation_wing_wall_length (defa	4.0000
foundation_wing_wall_width (defa	3.0000
parapet_bottom_angle (default)	2.00°
parapet_bottom_width (default)	0.5000
parapet_gap (default)	0.1000
parapet_height (default)	1.0692
parapet_top_angle (default)	1.00°
parapet_top_width (default)	0.5000
railing_height (default)	1.2000
railing_horizontal_space (default)	2.0000
railing_vertical_space (default)	0.3000
abutments_angle (default)	10.00°
retainingwall_crown_thickness (def	1.0000
retainingwall_height (default)	5.0000
retainingwall_length (default)	30.0000
retainingwall_support_width (defau	1.0000
slab_thickness (default)	1.0000
span_length (default)	10.0000
wingwall_bottom_length (default)	2.0000
wingwall_cap_height (default)	1.0000
wingwall_thickness (default)	1.0000
wingwall_top_length (default)	5.0000

Figure 5.2: 3D PPM of a single-span concrete bridge created following reverse engineering.

This 3D model is completely parametric and dependent on the value of parameters. It can be defined in most of the existing BIM-authoring tools. Following reverse engineering in CAD, this model requires a specific set of parameters. This user-dependent definition of the model restricts the problem space to a model that satisfies the engineering knowledge in the bridge design. In other words, the exact set of parameters is known, while the value of each parameter is not. If the value of these parameters can be derived from the segmented point clouds and injected into this model, it can update its shape and represent the geometric DT of the bridge. The only remaining question is how to derive the value of parameters that this 3D dummy model needs. In the upcoming chapter, we delve into addressing this question by introducing PPMs.

6 Automated Parametric Modeling[1]

Semantic segmentation of bridge point clouds leads to the partial point clouds of individual bridge elements. The next essential step in the geometric digital twinning method is to generate the volumetric model of the bridge components and assemble them to create the entire bridge model. To this end, solid modeling approaches mentioned in Section 3.2.1 can be employed. Parametric modeling is a modeling technique that not only generates the volumetric 3D models but also stores all the dependencies and relations among elements. It also provides an access point through which the bidirectional updates can be handled; one of the requirements of a geometric bridge DT is to stay connected with the physical asset.

Following the definition of parametric models and the requirements of bridge DTs, parametric modeling can be mentioned as a suitable solid modeling approach for generating geometric bridge DTs. However, parametric modeling is the most expensive approach, requiring more effort in the design and implementation phases. To handle this challenge, this section presents an automated framework, as shown in Figure 6.1, to generate the parametric model of existing bridges from their segmented PCD.

Reverse engineering with parametric modeling is a technique commonly used in the industry to convert scanned data to a CAD model. Following a reverse engineering approach with parametric modeling, PPMs are proposed to represent the bridge or the bridge component geometry. Reverse engineering proposes the desired final model to achieve at the beginning of the process, while parametric modeling keeps the model adjustable for the required reviews. Through these techniques, the initial model can be compared and become closer in shape to the scanned data by adjusting the value of parameters. PPMs are created based on a set of parameters as well as constraints and fed by analyzing the bridge point clouds.

[1]Significant parts of this chapter have been previously published in the Journal of Automation in Construction. The paper can be accessed at `https://www.sciencedirect.com/science/article/pii/S0926580523003618`.

PPMs are constant in type; however, their geometry can be adjusted/updated based on the input value of parameters. They are created purposefully to end up with the anticipated geometric DT model at the start of the process. Leveraging the parametric design of PPMs, a list of candidates is generated and adjusted through a local metaheuristic optimization to fit them into the point cloud of bridge elements. To assemble the fitted PPMs, the extracted parameters from the pieces are integrated through a global metaheuristic optimization. To generate the model of the entire bridge, the extracted parameter values are injected into the bridge's 3D PPM.

As a result, an inherently consistent geometric-semantic model is obtained that not only resembles the input bridge point cloud but also preserves all the relations and dependencies between the bridge components. The prospected benefit of this approach for end users is the massive reduction of the effort to create the geometric model of the bridge from PCD. Considering the desired model of the bridge, PPMs are designed in this section and used to extract the value of parameters from point clouds. The optimized PPMs are then assembled, and the resulting parameters are imported into the initial model to generate the parametric model of the entire bridge.

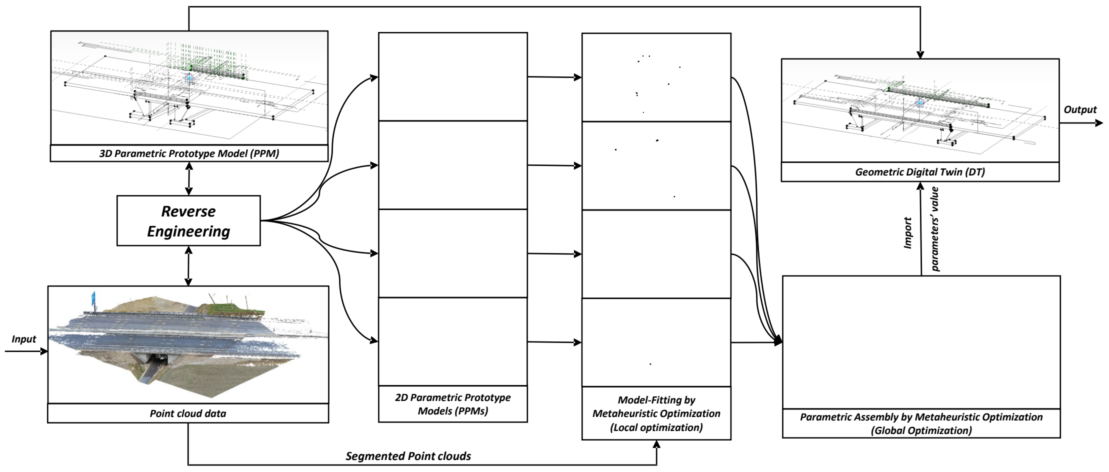

3D Parametric Prototype Model (PPM)
Reverse Engineering
Input
Point cloud data
2D Parametric Prototype Models (PPMs)
Model-Fitting by Metaheuristic Optimization (Local optimization)
Geometric Digital Twin (DT)
Output
Import
parameters' value
Parametric Assembly by Metaheuristic Optimization (Global Optimization)
Segmented Point clouds

Figure 6.1: Proposed method for geometric parametric modeling of bridges.

6.1 Parametric Prototype Models (PPMs)

A parametric model includes several parameters through which it can be altered. Also, it comprises a set of constraints that control and preserve the object's shape while being updated. In 3D modeling software, the parametric modeling process is started mainly by drawing 2D sketches on reference/working planes. These 2D sketches are refined and used by functionalities such as extrude, sweep, loft, and rotation to create a volumetric 3D model. Inspired by this process, we define a Parametric Prototype Model (PPM) as a dummy model comprising human-definable parameters and constraints that can update the shape. Figure 6.2 shows three typical 2D PPMs constructed by a set of parameters and constraints describing the geometric shapes. Parameters include the coordinate of origin, length of the edges, and angles, while geometric constraints might consist of horizontal, vertical, perpendicular, coincident, etc., constraints to restrict the geometry.

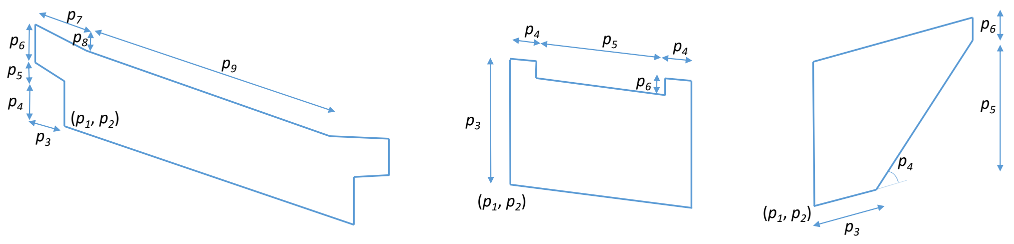


Figure 6.2: Various examples of 2D PPMs.

In particular, a PPM has three features. It contains a finite number of parameters and constraints, has a specific object type, and is a function of parameter values. For instance, a 2D rectangular PPM must be described with only four parameters, including the coordinate of origin (O_x, O_y), length, and width, as this object type has these parameters in the definition. It must also be a function of the parameter values, i.e., it can update its shape with new values of a parameter, such as width.

Contrary to the conventional model fitting methods, PPMs pave the way to fitting into not only the point cloud of simple geometries but also more complicated geometries that commonly exist in bridges. The programming process of a PPM is started from an origin and extended to other vertices based on the value of parameters. Concurrently, constraints such as parallelism, connectivity, perpendicularity, and symmetry are implicitly applied to the prototype model. Using Object-Oriented Programming (OOP) as an analogy, the PPM of an element is the instance of a class containing attributes such as dimensional values (i.e., parameter values) and constraints. Objects generated from the class will have different parameter values.

Figure 6.3 shows the PPM of a typical bridge deck described by a set of parameters. As can be seen, any change in the value of parameters leads to an instance of the bridge deck class with new dimensions. Considering a point cloud associated with this bridge deck, a list of candidates/solutions can be created and proposed for the value of dimensions

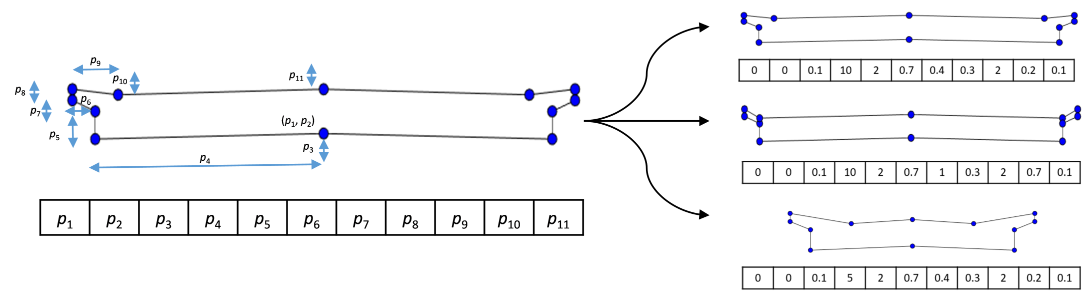


Figure 6.3: PPM of a typical bridge deck.

the point cloud represents. To determine the value of parameters through a PPM, each candidate needs to be quantified based on its similarity to the point cloud. To this end, a fitness function is defined in the next section which subsequently serves as the objective for a metaheuristic optimization algorithm.

6.2 Model-to-Cloud Fitting

A PPM is defined numerically based on a set of parameters and constraints. Therefore, the mathematical model of the PPM cannot be expressed and derived simply by a gradient-based algorithm. To address this issue, metaheuristic algorithms can be employed to adjust PPMs and fit them into the point cloud of elements. To instantiate a PPM, random values can be generated in predefined ranges inspired by bridge engineering knowledge. To fit a PPM, the shortest Euclidean distance of the edges to the point cloud must be minimized.

Considering a set of points $\mathcal{S} = \{s_i | i = 1, ..., n\}$, where $s_i \in \mathbb{R}^2$, and a 2D PPM described by a set of parameters $\mathcal{X} = \{x_r | r = 1, ..., m\}$ with the lower bound l_r and upper bound u_r, in which $x_r \in [l_r, u_r]$, the following objective/fitness function can be defined in the term of mean absolute error (MAE):

$$F(x_1, ..., x_m) = \frac{1}{n} \sum_{i=1}^{n} e_i, \tag{6.1}$$

where e_i is a positive value describing the shortest distance of the i^{th} point to the edges and vertices of the PPM.

A PPM can typically have any position with respect to points set in the space of the problem. The aforementioned function is capable of minimizing the distance of points to the edges of the PPM. However, it cannot guarantee all the edges are fitted into the point cloud. This is because some edges might not find any point in their vicinity. Thus, no point exists to apply a value of error to such edges, and the corresponding parameters to the location of these edges cannot be adjusted during the optimization process. In other words, these edges have a redundant degree(s) of freedom that must be closed.

This case is even intensified in occluded point clouds in which some parts are empty of points. To improve the performance of the optimization algorithm and enable it to handle occlusion, the concept of *active* and *passive* edges is proposed.

Definition: An edge is called *active* when it has at least one of the following conditions: 1. It possesses at least two points, or 2. It possesses at least a point and has a slope constraint. In any other conditions, the edge is called *passive* as it does not have enough points or constraints to contribute to the optimization process.

To activate the passive edges of a PPM with the number of k edges, a new penalty term $(\lambda_j e'_j)$ is defined for each edge j and added to the previous fitness function as follows:

$$F(x_1, ..., x_m) = \frac{1}{n}\sum_{i=1}^{n} e_i + \frac{1}{k}\sum_{j=1}^{k} \lambda_j e'_j, \tag{6.2}$$

where λ_j is a binary value controlling the activity of edges, i.e., 0 for active edges and 1 for passive edges, and e'_j is the value of error required to activate the passive edges.

Considering the shortest distance of points to the edges, subsets of points can be created and assigned to each edge. Thus, the first term of the fitness function can be rewritten for the edges, and the following simplified fitness function is achieved:

$$F(x_1, ..., x_m) = \frac{1}{k}\sum_{j=1}^{k}(e_j + \lambda_j e'_j), \tag{6.3}$$

where e_j is the total distance of the edge j to its nearest points.

To determine whether an edge is active or passive, the number of points assigned to the edge must be counted during the optimization process; this number might vary as the PPM moves onto the plane and updates its shape. Also, the slope constraints of the edge, such as vertical and horizontal constraints, must be controlled; these constraints are constant. Using such information and considering the definition of the active edges, the passive edges can be detected and activated.

To activate a passive edge, the two neighboring edges of the passive edge are considered, and the value of e'_j is calculated accordingly. Figure 6.4 shows a PPM with four edges in different model-fitting scenarios. Assume the edges of the PPM have been assigned the index $j = \{1, 2, 3, 4\}$ from the left edge in clockwise order. Figure 6.4a depicts a rectangular PPM as all the edges of the PPM have horizontal or vertical constraints. As can be seen, there is a point close to each edge of the PPM; thus, the edges possess a point. Considering the relative position of points with respect to the edges and constraints controlling the slope (one point and a slope constraint), all the edges are active, and the value of error is only the shortest distance of edges to the points, i.e., no additional value of error (penalty) is required to be added ($\lambda_j = 0$). Figure 6.4b shows another scenario in which the left edge has no point and only has a vertical constraint. Since this edge has only a constraint and no point, it cannot be involved in the optimization process (a passive edge).

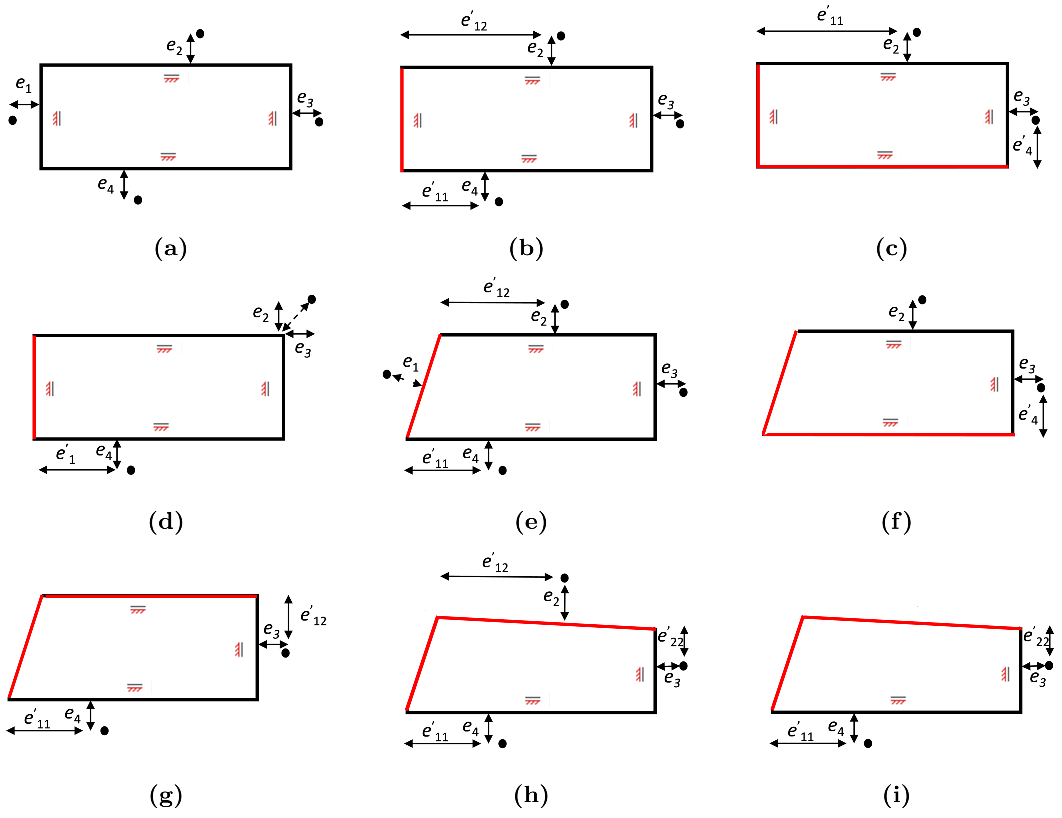


Figure 6.4: Various scenarios of fitting a typical PPM into a set of points.

To activate this edge and close its translational degree of freedom, a single point needs to be assigned to this edge from the neighboring edges to meet the condition of one point and a slope constraint. As both neighboring edges have a point and the edge has a vertical constraint, a value of error equal to the minimum distance of the left edge (passive edge) to the closest point of the neighboring edges is added ($e'_1 = min(e'_{11}, e'_{12})$). Figure 6.4c illustrates another case in which the bottom and left edges are passive. However, they both have at least a neighboring edge with a point through which they can be activated (one point and a slope constraint). In Figure 6.4d, the left, top, and right edges have no points; however, the bottom edge, the right endpoint of the top edge, and the top endpoint of the right edge possess a point. In this case, the point belonging to the endpoints can activate the corresponding edges, i.e., the top and right edges are still active. Nonetheless, this point cannot be used for activating the neighboring edges. Thus, the left edge is only activated based on the point belonging to the bottom edge.

Figure 6.4e demonstrates a PPM in which the left edge has no slope constraint. Even though this edge possesses a point, it is still passive, as it needs one more point to satisfy the condition of two points. This edge can be activated by adding the mean value of errors ($e'_1 = mean(e'_{11}, e'_{12})$). Figure 6.4f also illustrates a PPM in which the left and bottom edges are passive due to a lack of points. Although the bottom edge has a constraint and only a point from the neighboring edge is sufficient to activate it, the left edge has neither a constraint nor two points from each neighboring edge to reach the activity condition of two points. Thus, model fitting is impossible in this case, as the slope of the left edge cannot be recognized. As a result, a high error value ($e'_1 = 10e^3$) is added to decrease the selection probability of this PPM.

Figure 6.4g illustrates a PPM whose left and top edges are passive. Even though the top edge only needs a point to reach the condition of a point and a constraint, the left edge cannot find two points from each neighboring edge to become activated. In the next case (Figure 6.4h), the left and top edges both have no constraint, and they are passive. The top edge already has a point, and it needs only a point from the neighboring edges to be activated. The left edge, however, has no point and needs two points, each one from a neighboring edge. As can be seen, the neighboring edges can give a point to this edge; thus, it can be activated as well. The last case shown in Figure 6.4i is similar to the previous case, while the left and top edges can only take a point from one of the neighboring edges. Therefore, model-fitting, in this case, is impossible as well.

While Equation 6.3 is capable of model-fitting and handling occlusion to a large extent, it cannot ensure the equal contribution of edges to the optimization process. The current definition of the objective function is based on the distribution of points across the edges of the PPM. This distribution might vary from a slight bias to a severe imbalance where some edges have one point, and others have hundreds of points. This results in a lack of sensitivity to the movement of edges with a lower number of points. To address this challenge, the weighted summation of errors resulting from each edge is calculated.

Considering a point cloud with n points and a PPM with a number of k edges whose edge j possesses t_j points, the edge weight ω_j can be calculated as follows:

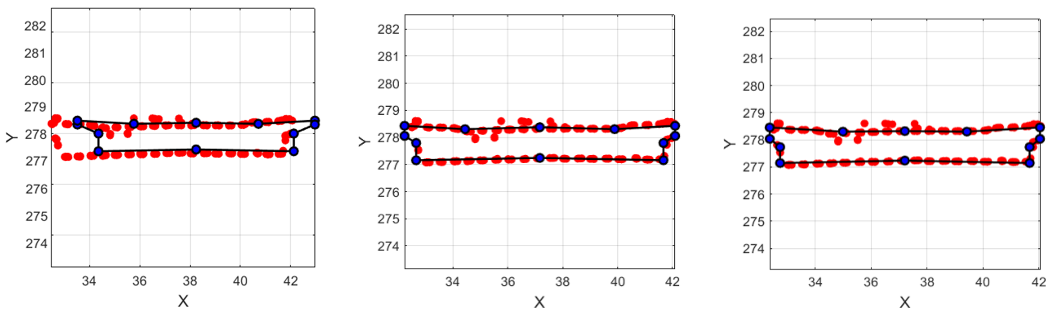
$$\forall j : 1 \leq j \leq k \quad \rightarrow \quad \alpha_j = \frac{t_j}{n} \quad \& \quad \omega_j = \frac{1}{\alpha_j + \beta}, \tag{6.4}$$

where $1 \leq t_j \leq n$, $\sum_{j=1}^{k} t_j = n$, and β is a constant value (0.02) preventing a zero denominator.

The weighted fitness function of the problem can also be rewritten, and an optimization/minimization problem is defined for fitting a PPM into a point cloud as below:

$$\begin{aligned} &\textit{To minimize:} \quad F(x_1, ..., x_m) = \frac{1}{k}\sum_{j=1}^{k} \omega_j (e_j + \lambda_j e'_j), \\ &\textit{Subjected to:} \quad l_r \leq x_r \leq u_r \end{aligned} \tag{6.5}$$

After the initialization process, a list of candidates (population) is randomly generated from a PPM by a metaheuristic optimization algorithm. This list will be then improved by adjusting the initial value of parameters and minimizing the value of error resulting from Equation 6.5. As can be seen in Figure 6.5, this optimization process leads to a PPM that resembles the input point cloud, and its value of parameters is a close approximation of the values the point cloud represents.



(a) **(b)** **(c)**

Figure 6.5: PPM of a typical bridge deck during the optimization process: (a) iteration 1; (b) iteration 20; (c) iteration 100.

The approach presented here provides an element-wise model-fitting, i.e., each PPM can extract the value of parameters from a single component (face/cross-section). In the next section, a global optimization problem is defined to assemble and integrate all the pieces and create the parametric model of the entire bridge.

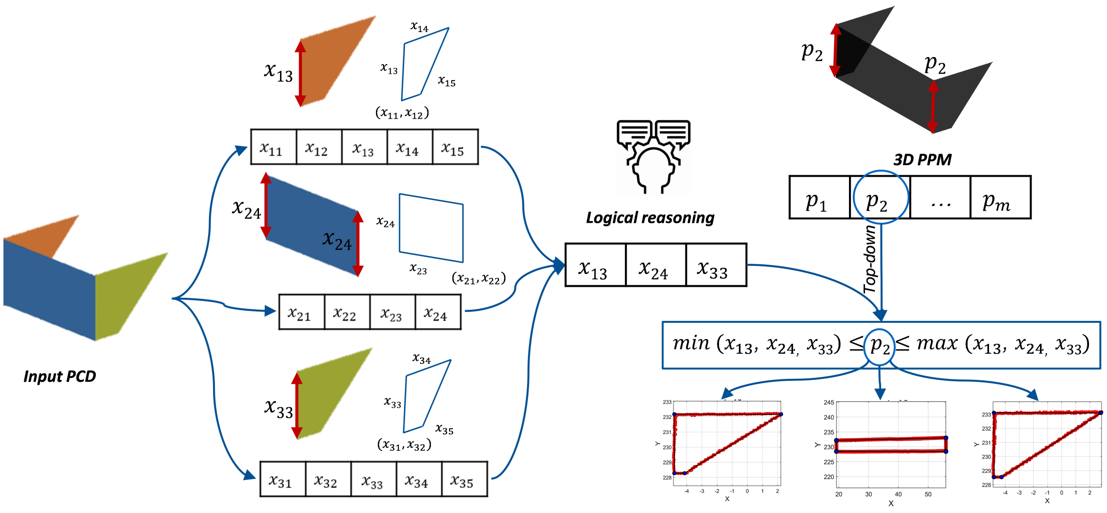
6.3 Parametric Assembly

The model-fitting process through PPMs leads to a list of parameters representing the point cloud of elements. To create the parametric model of the entire bridge, these components must be assembled consistently, e.g., dimensions of shared edges and faces must be equal.

For this purpose, snapping algorithms have been generally proposed to connect and integrate pieces [218, 219]. These algorithms discover matches between polygons and search for adjacent vertices considering various conditions. The neighboring vertices are then replaced with a new vertex representing all the vertices. Snapping algorithms can be practical for model reconstruction and 3D representation of bridges. However, the model cannot stay parametric in those algorithms as the location of vertices is a function of parameters, and this function needs to meet a set of constraints. Furthermore, snapping algorithms generally follow a bottom-up approach, starting from vertices and edges, and mostly require setting problem-specific thresholds. To handle this challenge, a top-down approach is proposed, and a global optimization problem is defined to assemble the bridge components.

Figure 6.6 illustrates the point cloud of an abutment comprising two wing walls and a retaining wall. Following the proposed method in Section 6.2, a set of parameters can be obtained for each face/cross-section by solving element-wise optimization problems associated with the 2D PPMs. Herein, the value of parameters has been shown by x_{ij}, where i and j are indices devoted to the face and parameter number, respectively. In a parametric assembly problem, sets containing common parameters among components can be found that logically need to be represented by a single parameter. For instance, $A_2 = \{x_{13}, x_{24}, x_{33}\}$ is a set including the values of height resulting from the initial model-fitting process. Considering a top-down approach, the 3D PPM of an abutment can be created with a group of unique parameters, among which there is only a single parameter, such as p_2 controlling the height of the abutment. To integrate PPMs, a representative value must be generated from the set A_2 and applied to the parameter p_2. Although averaging the set A_2 can provide a single representative value, it cannot lead to a permanent solution.

This results from the fact that a parametric model generally contains complicated dependencies and relations, and it is not apparent how the dependent parameters are affected by the average function. Considering the results of the initial element-wise model-fitting, each member of the set A_2 can be a proper candidate for the parameter p_2. The discrete set A_2 can be converted to a continuous interval by using the min and



Input PCD
x_{13}
x_{11} x_{12} x_{13} x_{14} x_{15}
x_{24}
x_{21} x_{22} x_{23} x_{24}
x_{33}
x_{31} x_{32} x_{33} x_{34} x_{35}
Logical reasoning
x_{13} x_{24} x_{33}
3D PPM
p_1 p_2 … p_m
Top-down
$min(x_{13}, x_{24}, x_{33}) \leq p_2 \leq max(x_{13}, x_{24}, x_{33})$

Figure 6.6: Assembly process of a typical abutment.

max functions, and each value in this range is considered a possible value for p_2 as well ($min(A_2) \leq p_2 \leq max(A_2)$).

Conversely, the value of the parameter p_2 should apply to the PPMs associated with the set A_2 and still retain them as close as possible to their corresponding point clouds. To satisfy these conditions, random values of the parameter p_2 can be generated in the interval resulting from the initial model-fitting, and their impact is evaluated on all the involved PPMs. In doing so, the value leading to the best fitting of all the PPMs can be approximated. This top-down method is only dependent on the proposed list of candidates for a parameter. This example can be extended and is expressed as an optimization problem for the parametric modeling of the entire bridge.

Let $\mathcal{X} = \{x_i | i = 1, ..., n\}$ be the set of all the possible parameter values resulting from fitting several PPMs into their corresponding point clouds. Following the reverse engineering (top-down) approach, assume $\mathcal{P} = \{p_j | j = 1, ..., m\}$ is also the target set of parameter values required to create the parametric model of the entire bridge. Considering the label of parameters, the initial set $\mathcal{X}$ can be divided into smaller sets of parameters that need to be assembled. Thus, a family of sets is obtained $\mathcal{A} = \{A_j | j = 1, ..., m\}$, where $A_j \subseteq \mathcal{X}$ and contains all the possible candidate values for the corresponding parameter p_j. The parametric assembly process of the number of h PPMs can be described as an optimization/minimization problem as follows:

$$\begin{aligned} &\textit{To minimize:} \quad G(p_1, ..., p_m) = \frac{1}{h}\sum_{v=1}^{h} \omega_v F_v, \\ &\textit{Subjected to:} \quad min(A_j) \leq p_j \leq max(A_j) \end{aligned} \tag{6.6}$$

where F_v is the fitness function described in Equation 6.5 and ω_v is the weight assigned to each PPM to balance the model-fitting errors. The value of ω_v can be calculated using Equation 6.4 based on the total number of points and the number of assigned points to each PPM.

This objective function receives a set of parameter values, randomly generated in ranges obtained by the initial element-wise model-fitting. It adjusts all the involved PPMs and fits them into the point cloud of the entire bridge.

6.4 Selection of PPMs

Given the point cloud of a bridge component (face/cross-section), a proper PPM needs to be selected to describe the input sample. For instance, the PPM of a bridge deck cannot be used to derive the parameter values from an abutment point cloud as these elements are of different types.

To address this problem, a library/catalog of bridge elements is created in which various types of PPMs exist. To select the appropriate PPM, the similarity of the input point cloud to all the PPMs is checked. For this purpose, two methods, called supervised and unsupervised selection, are proposed to determine the PPM required for model fitting. As shown in Figure 6.7, both of the methods are classifiers, however, with different levels of supervision.

The supervised selection method requires a machine/deep learning model to be trained on the point cloud of the existing bridge elements in the catalog. There are many models in the literature that can be used as a point cloud classifier [155, 153, 156, 157]. The trained model can receive the point cloud of bridge elements and determine the type of PPM required for model fitting. This approach needs a large dataset of point clouds as well as an annotation process. However, the trained model can instantly select and call the appropriate PPM from the catalog.

The unsupervised selection method fits each existing PPM in the library/catalog to the input point clouds by solving a piece-wise optimization problem. Each model-fitting process leads to a value of model-fitting error describing the similarity of the input point cloud to the PPM. At the end of the process, the PPM with the lowest value of error is selected as it is more likely to represent the input point cloud. In comparison with the supervised selection, this method does not require a dataset for training and can directly classify the point cloud of bridge components. However, testing each PPM on the input point cloud requires more time. The supervised and unsupervised selection methods can both be used interchangeably for the selection of PPMs.

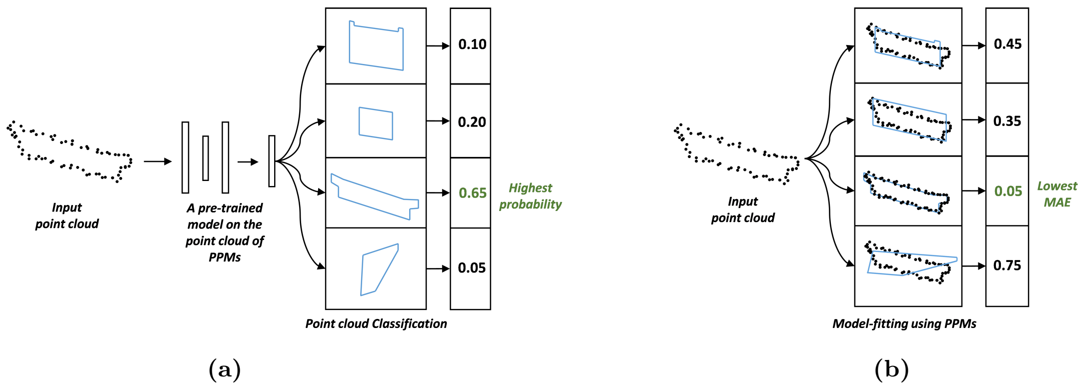


Figure 6.7: Selection of PPMs based on the input point cloud: (a) supervised selection; (b) unsupervised selection.

6.5 Selection of Metaheuristic Algorithm

Various metaheuristic algorithms can be used for fitting PPMs into point clouds. To evaluate the impact of the algorithms on the performance of the model, ten different metaheuristic algorithms, including Particle Swarm Optimization (PSO) [96], Genetic Algorithm (GA) [97], Harmony Search (HS) [220], Differential Evolution (DE) [221], Invasive Weed Optimization (IWO) [222], Shuffled Frog Leaping Algorithm (SFLA) [223], Teaching Learning Based Optimization (TLBO) [98], Firefly Algorithm (FA) [100], Simulated Annealing (SA) [224], and hybrid PSO-GA [225] are tested.

Each algorithm is run ten times to fit an I-shaped beam PPM into a point cloud, and the resulting mean convergence diagrams, as well as the average time required for model fitting, are presented. The hyperparameters of each algorithm have been tuned such that the best results are achieved for a specific number of iterations in a reasonable time interval. Figure 6.8a shows the obtained convergence diagrams from the metaheuristic algorithms in a logarithmic scale. As can be seen, three algorithms of PSO-GA, TLBO, and FA have been capable of gaining the lowest model-fitting errors, respectively. Figure 6.8b also illustrates the average required time for fitting the PPMs in which the HS algorithm has achieved the lowest modeling time.

Comparing the results of PSO-GA, TLBO, and FA in terms of time demonstrates the faster performance of TLBO in the model-fitting task. Among these three algorithms, TLBO only needs one hyperparameter (number of particles/population) and stopping criteria, while the two other algorithms have more problem-dependent hyperparameters for tuning. Therefore, TLBO can provide a higher level of automation with minimal user intervention. Considering the algorithm's stability in convergence, the required time for model-fitting, and the number of hyperparameters, TLBO is selected while all other algorithms can also be utilized.

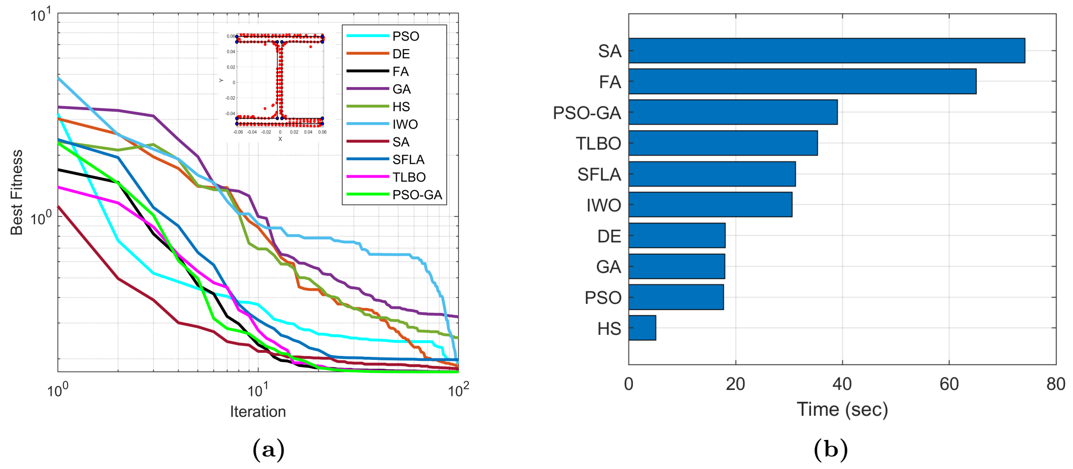


Figure 6.8: Comparing the performance of 10 different metaheuristic algorithms in a PPM-to-cloud fitting task: (a) Convergence diagram; (b) Convergence time.

7 Results & Case Studies

This chapter presents the results and case studies by starting from the raw point cloud captured from real bridges and creating the geometric DT of each sample. In the first part (Section 7.1), the process of generating the dense point clouds, including the acquisition process as well as the point cloud generation, is explained, and an overview of the samples is presented. In Section 7.2, the results of applying MSFD-Net and RandLA-net [158] to the point cloud of the bridges are reported, and these models are compared using various evaluation methods and in terms of different statistical metrics. This chapter continues to Section 7.3, where the required algorithms to prepare the segmented point clouds for generating bridge models are addressed. In Section 7.4, the proposed parametric modeling approach is applied to the point clouds, and the 3D geometric model of bridges is generated. This section also models bridge components of multi-span bridges to show the application range of the proposed method. It further discusses the algorithm's advantages over other existing methods in the literature and compares it with them.

7.1 Dataset

The point cloud of ten single-span reinforced concrete (RC) highway bridges in Bavaria, Germany, is used for evaluation and model reconstruction. This dataset has been acquired through aerial photogrammetry by flying a drone around the structure and underneath the bridge deck to take photographs from various angles to meet a minimum 75% frontal and 60% side overlap. All the captured images have the same resolution of 5472 $\times$ 3078. This dataset has been processed by Agisoft [226]. This commercial software has three major steps to generate a dense point cloud. The first step loads the photographs and specifies their location following the GPS information. Next, images are aligned; this step involves detecting and matching key points between the overlapping images and estimating the camera parameters (position and orientation) for each image. Agisoft employs advanced feature detection algorithms to identify key points within images, align them, and generate 3D point clouds. The algorithms include the Scale-Invariant Feature Transform (SIFT) and Speeded Up Robust Features (SURF), which detect distinctive points that remain invariant under various transformations such as scale, rotation, and illumination changes. Once key points are detected in each image, Agisoft matches corresponding key points across multiple images, using techniques such as descriptor comparison and outlier rejection. The software then conducts bundle adjustment to refine camera positions and orientations, optimizing them to minimize reprojection errors and ensure precise alignment. Through this process, detailed and accurate 3D models can be generated from the collected 2D images. Additionally, it utilizes a depth map or multi-view stereo algorithm to further enhance the density of the point cloud by estimating the 3D positions of points in the scene and refining depth information. In this research, the default setting of 40,000 key points for the Key Point Limit appeared well-suited as no significant enhancement in the quality of the point cloud could be observed when employing an alignment that exceeds this key point threshold. In cases where the algorithm could detect no similar key point between the images, the key points were registered manually. All the dense point clouds have been generated following High or Ultra High quality settings. This quality measure determines the desired quality of the depth map generation. Higher quality settings can lead to point clouds with more detailed and accurate geometry, while they require a longer time for processing. For instance, ultra high setting means the processing of original photos, while each following step implies preliminary image size downscaling by a factor of 4 (2 times by each side).

Figure 7.1 shows the generated point cloud samples for the bridges. All the bridge samples have the same type, i.e., single-span RC bridges with a straight bridge deck supported by two abutments. However, they have different sizes with dimension values. To decrease the processing load of the algorithms, all the bridge samples have been subsampled by the uniform grid subsampling method with a grid size of 5 cm. This step led to point clouds with an average density of 252 points/m^2 and around 2 million points per sample. This means the point clouds are still large-scale as they contain more than a million points for processing.

In addition to the main dataset used for modeling single-span bridges, other datasets such as the Cambridge bridge dataset [227] are used for validating the method on bridge components of multi-span bridges. Contrary to the first samples, they are laser scanning PCD, which can prove the method can not only process photogrammetric PCD but also laser scanning. Further details about the datasets are presented in the next sections.

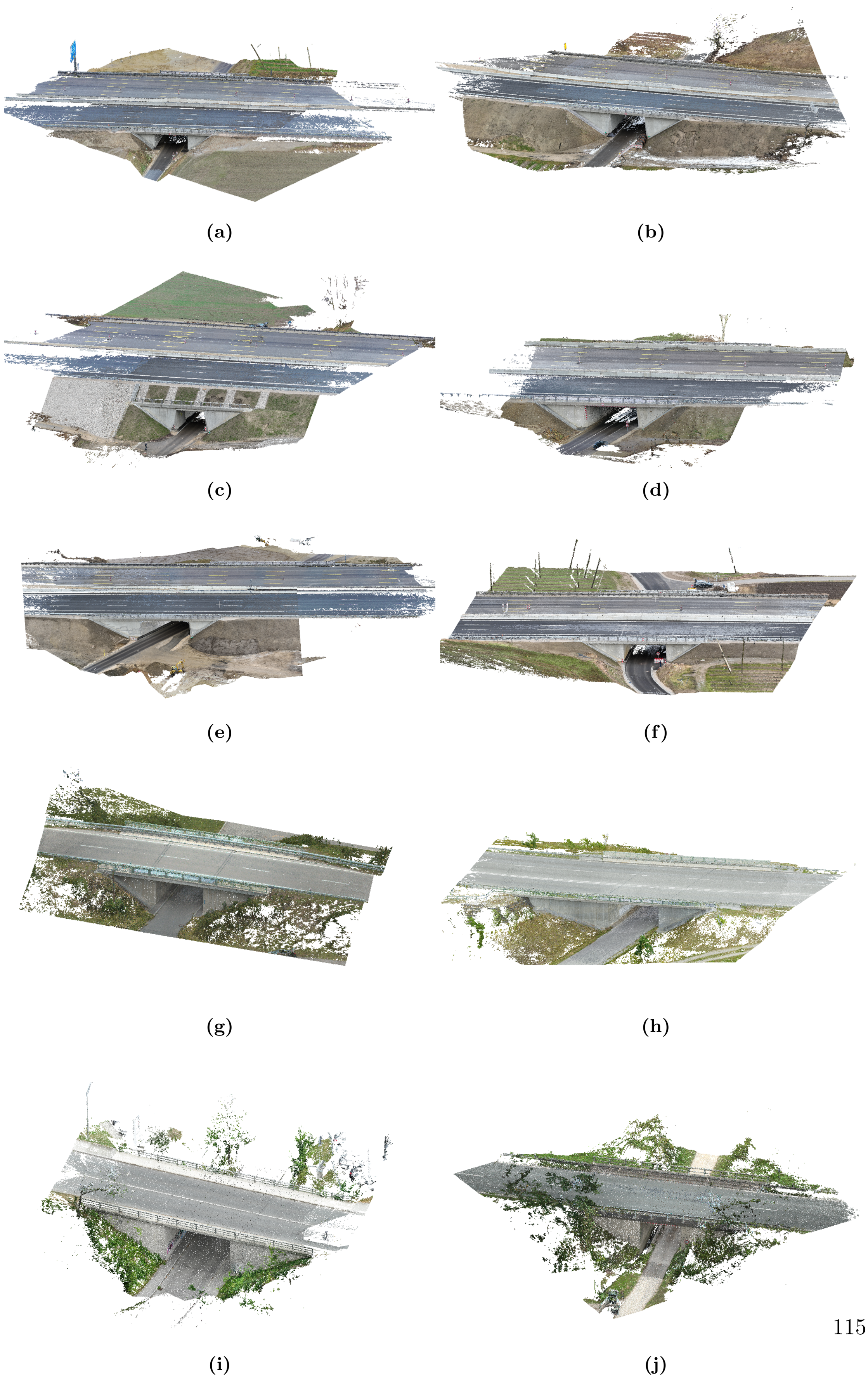


Figure 7.1: Photogrammetric PCD of ten single-span RC bridges. (a-j) show the bridge samples 1-10, respectively.

7.2 Semantic Segmentation

This section covers the semantic segmentation of bridge point clouds using the proposed model (MSFD-Net) in Chapter 4. It starts with discussing the preparation of the dataset (annotation) and introduces the validation methods for testing the model. It also elaborates on the hyperparameters of the model and the values used in the training phase. Finally, it presents the performance tests of MSFD-Net as well as RandLA-Net [158] in the prediction of class labels in terms of different statistical metrics.

7.2.1 Data Preparation

The raw bridge point clouds generally contain (x, y, z) coordinates and RGB color codes. They might also contain other point-level features such as normal, curvature, roughness, etc. To generate the 3D model of single-span RC bridges, their point clouds can be segmented into four classes: background, deck, abutment, and railing. Detecting these elements in the point cloud streamlines the parametric modeling process to a large extent. Hence, these four classes are considered for semantic segmentation.

Data loader of 3D DL models load point clouds in several specific formats that have been mostly inspired by the data structure of benchmark datasets such as S3DIS [228], SemanticKitti [229], Toronto3D [230], etc. The data loader of MSFD-Net is capable of reading point clouds in all those formats. In this research, the bridge point clouds have been prepared in the S3DIS format as it simply needs the text (.txt) file of the classes as well as the entire point cloud of each sample. The annotation process of point clouds is similar to the manual segmentation process followed in current practice (Section 2.6). The points can be labeled using a brushing tool or by drawing polygons around elements and separating them.

To process the point clouds with MSFD-Net, normals need to be calculated. Normal vectors are only used in the LSuR module of the model to calculate local features. If normals have not been produced during the point cloud generation phase, they can be calculated using a data structure such as kd-tree and a search algorithm like KNN. MSFD-Net is invariant to the direction of normal vectors; thus, they can be calculated with any arbitrary direction depending on a viewpoint.

All the bridge samples are translated to the origin of the coordinate system. The RGB color codes are normalized in the range of [0, 1] by dividing the values by 255. However, the (x, y, z) coordinate of points is not normalized as this step can lead to a large difference in the samples' scale, especially in noisy point clouds. Linear normalization functions normalize point clouds following the Axis-Aligned Bounding Box

(AABB) surrounding the point cloud. The AABB is affected by noise points, leading to large differences between bridge samples after normalization. To augment the point cloud samples, all the samples are randomly rotated around the z-axis. Furthermore, independent noise is added to each point by jittering the samples with the amount of 1 mm. All these steps are conducted prior to the training process, ensuring the same samples are being used for training both models. Due to the different number of points in each class (imbalanced dataset), class weights are computed based on the number of points over all the training samples in each class. Subsequently, a higher weight is assigned to the classes with a lower number of points and vice versa.

7.2.2 Validation Method

To evaluate the model's performance, expensive tests are conducted on MSFD-Net and RandLA-Net [158] simultaneously through the leave-one-out cross-validation (LOOCV). LOOCV is the most expensive k-fold cross-validation method in which the value of k is set to the number of samples. Each fold holds one sample for testing, and the models are trained on the remaining samples. In this research, there are ten bridge samples in total; thus, the value of k is set to ten, and in each try, the model is trained on nine samples and tested on one remaining unseen sample. Finally, the results are presented as the average/mean value obtained from all the tests. Various metrics are reported for each bridge sample and its classes. The Overall Accuracy (OA) of bridge point classification can be calculated as follows:

$$OA = \frac{TP + TN}{TP + TN + FP + FN} \tag{7.1}$$

where TP, TN, FP, and FN are the total number of true positive, true negative, false positive, and false negative predictions by the model, respectively. They can be calculated from the confusion matrix of the model after classifying points.

The performance of the model in prediction can also be described in terms of Recall, precision, F1 score, and intersection over union (IoU) for each class i as below:

$$Recall_i = \frac{TP_i}{TP_i + FN_i} \tag{7.2}$$

$$Precision_i = \frac{TP_i}{TP_i + FP_i}, \tag{7.3}$$

$$F1 = \frac{2TP_i}{2TP_i + FP_i + FN_i}, \tag{7.4}$$

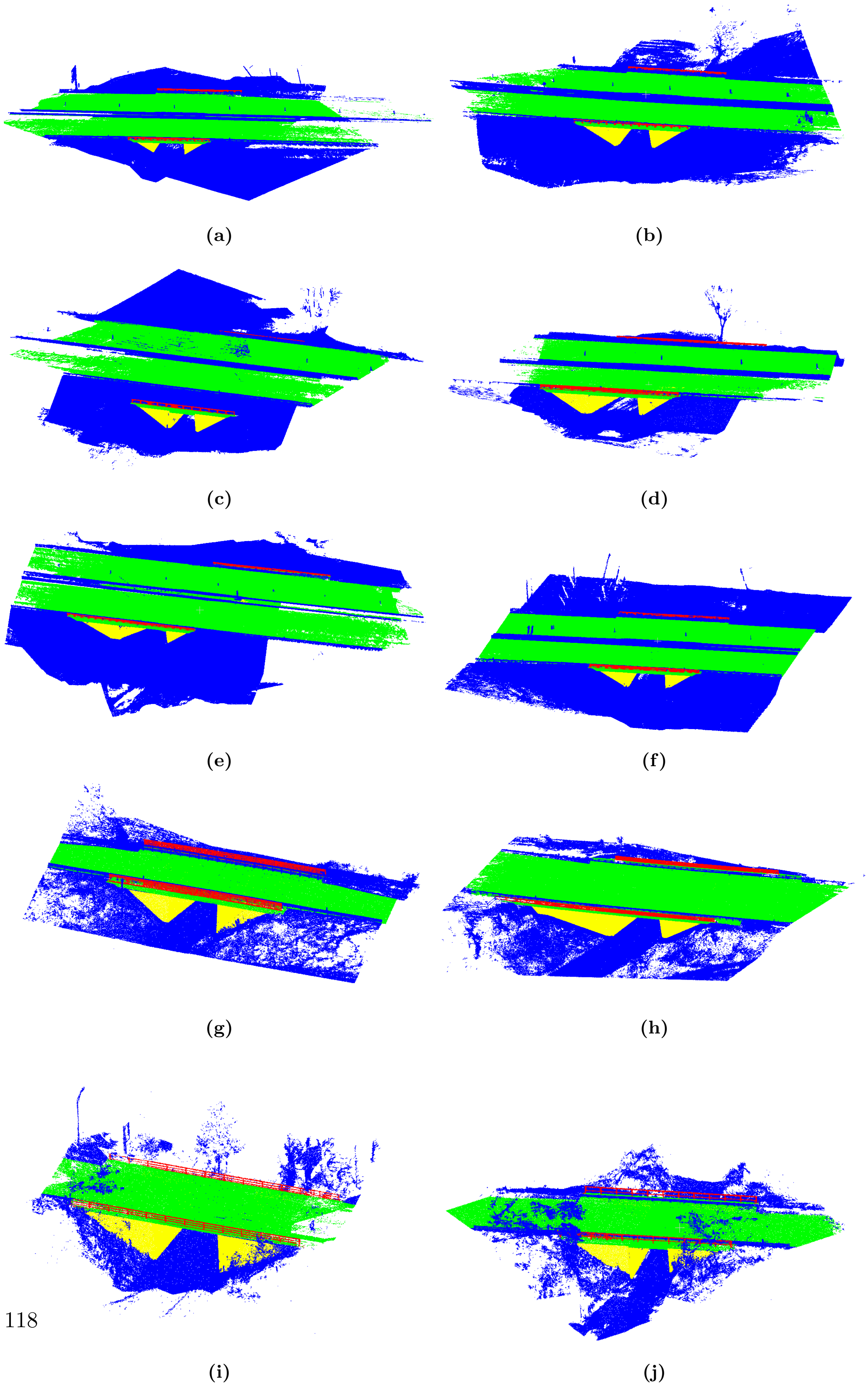


Figure 7.2: Annotated PCD of bridges. (a-j) show the bridge samples 1-10, respectively. Class colors: Blue: Background, Green: Deck, Railing: Red, Abutment: Yellow.

$$IoU_i = \frac{TP_i}{TP_i + FN_i + FP_i}, \tag{7.5}$$

The mean (average) performance of the models can be summarized through mean recall (mRecall), mean precision (mPrecision), mean F1 score (mF1), and mean intersection over union (mIoU). These metrics can be calculated as follows:

$$mRecall = \frac{1}{N} \sum_{i=1}^{N} Recall_i, \tag{7.6}$$

$$mPrecision = \frac{1}{N} \sum_{i=1}^{N} Precision_i, \tag{7.7}$$

$$mF1 = \frac{1}{N} \sum_{i=1}^{N} F1, \tag{7.8}$$

$$mIoU = \frac{1}{N} \sum_{i=1}^{N} IoU_i, \tag{7.9}$$

where N is the number of classes.

7.2.3 Hyperparameters

MSFD-Net and RandLA-Net both require a set of hyperparameters to be configured. The values of hyperparameters are determined through a trial and error process, and the values leading to the best performance are presented. Contrary to RandLA-Net, MSFD-Net grows not only in depth but also in the number of scales. Also, the spatial relationships between points are encoded in each scale. The number of encoding blocks (or scales) is the initial hyperparameter that needs to be determined. This research uses four encoding blocks/scales for MSFD-Net. The number of 16, 64, 128, and 256 features are extracted in each PFD module with two repetitions to increase the receptive field of points.

We configure RandLA-Net with its original architecture containing five encoding blocks from which the number 16, 64, 128, 256, and 512 features are extracted, respectively. Similarly, the encoding process is repeated twice in each encoding block. Four subsampling layers with a ratio of 1/4 are considered for the first four encoding blocks so that only 25 % of the points in each layer are retained. The last encoding layers are only subsampled with a ratio of 1/2 to emphasize fine features generated from that layer.

MSFD-Net only employs the first four subsampling layers with a ratio of 1/4. Sixteen neighbors are selected for the KNN algorithm, and eight features are adopted for the input layer. A batch size of 2 with a maximum size of 60,000 points with 100 steps per epoch is considered to train the models. The Adam optimizer is used to train the models with an exponential learning rate of 0.001 and a scheduler gamma value of 0.99 for 200 epochs. The hyperparameters of the models can be seen in Table 7.1.

Table 7.1: Hyperparameters of RandLA-Net and MSFD-Net.

Hyperparameter	RandLA-Net	MSFD-Net
Subsampling ratio	[4, 4, 4, 4, 2]	[4, 4, 4, 4]
Dim features	[16, 64, 128, 256, 512]	[16, 64, 128, 256]
Num neighbors	16	16
Max number of points	60,000	60,000
Batch size	2	2
Learning rate	0.001	0.001
Epochs	200	200
Scheduler gamma	0.99	0.99
Training steps-per-epoch	100	100
Validation steps-per-epoch	10	10

7.2.4 Performance Evaluation

MSFD-Net and RandLA-Net are trained on a single GPU (RTX 3080) with 16 GB RAM. The performance of the models is tested on exactly the same unseen samples through the LOOCV method. Also, the same augmentation methods are applied to the models for a reasonable comparison. The performance of each model is reported using the evaluation metrics introduced in Section 7.2.2, and the models' predictions are visualized.

MSFD-Net and RandLA-Net are trained each across ten folds, with nine samples assigned for training, while the performance is evaluated on the one remaining unseen sample. The same hyperparameter values (Table 7.1) are employed for all folds, ensuring no change in the value of hyperparameters is made moving from one fold to another. Additionally, both models are controlled to utilize identical point cloud samples throughout their training and testing phases, ensuring that augmentation does not cause significant variations in the dataset samples.

Table 7.2 demonstrates the results of RandLA-Net in terms of the evaluation metrics throughout the LOOCV tests. Each row shows the bridge sample tested after training on the other nine samples. Columns also represent the OA of the models as well as the mean values of Precision, Recall, F1 Score, and IoU obtained by averaging over classes. Additionally, the last row of the table illustrates the final results of LOOCV after testing all bridge samples based on each evaluation metric. As can be seen, RandLA-Net has been capable of achieving an OA of 96.15%. Moreover, it demonstrates good performance across multiple metrics with mPrecision, mRecall, mF1 Score, and mIoU values of 90.70%, 95.58%, 92.76%, and 87.20%, respectively. The results highlight the efficiency of RandLA-Net across bridges. However, variations in performance metrics across different bridges imply instability in the model's performance in the semantic segmentation of bridge point clouds. For instance, the value of mIoU ranges from 81.83% in Bridge 08 to 91.05% in Bridge 07, meaning a large difference (9.22%) in performance.

Table 7.2: Cross-validation results of RandLA-Net on ten bridges (%).

Samples	OA	mPrecision	mRecall	mF1	mIoU
Bridge01	95.33	91.53	96.96	94.00	88.87
Bridge02	96.79	88.12	97.97	92.01	86.30
Bridge03	94.87	87.03	95.37	90.54	83.66
Bridge04	96.73	92.11	98.03	94.78	90.27
Bridge05	96.35	92.14	96.59	94.22	89.28
Bridge06	97.90	90.22	98.65	93.80	89.00
Bridge07	97.88	92.74	98.24	95.20	91.05
Bridge08	94.66	89.65	88.14	88.78	81.83
Bridge09	95.20	92.99	92.29	92.42	86.25
Bridge10	95.81	90.45	93.54	91.83	85.54
Avg.	96.15	90.70	95.58	92.76	87.20

In comparison, Table 7.3 presents the performance evaluation of MSFD-Net in the context of the LOOCV test. As can be seen, MSFD-Net demonstrates a more promising performance, surpassing RandLA-Net in metrics. It achieves an OA of 96.97%, highlighting its superior overall accuracy. While the models' OA are close, performance differences become apparent in other metrics. MSFD-Net exhibits higher mean precision and recall than RandLA-Net, indicating a more robust capacity for precise positive predictions and accurate identification of relevant classes. It also tends to have higher F1 scores and IoU averages, indicating a better balance between precision and recall and improved spatial overlap prediction than RandLA-Net.

Furthermore, MSFD-Net demonstrates robustness in its predictive capabilities with mean Precision, Recall, F1 Score, and IoU values of 94.59%, 96.63%, 92.76%, and 91.57%, respectively. To illustrate the improvement that MSFD-Net brings over RandLA-Net, the percentage differences in their average performance metrics can be measured. MSFD-Net illustrates a 1.35% enhancement in OA compared to RandLA-Net. Moreover, it shows substantial improvements in mPrecision, exhibiting a remarkable 4.29% increase and a 1.10% enhancement in mRecall. The mF1 score, a crucial measure of the balance between mPrecision and mRecall, shows an improvement of 2.92% with MSFD-Net. Additionally, MSFD-Net significantly boosts the mIoU by 4.88%, indicating superior spatial overlap prediction capability compared to RandLA-Net.

Table 7.3: Cross-validation results of MSFD-Net on ten bridges (%).

Samples	OA	mPrecision	mRecall	mF1	mIoU
Bridge01	97.53	94.26	98.35	96.18	92.75
Bridge02	97.34	92.40	96.85	94.47	89.90
Bridge03	97.68	93.94	97.73	95.72	92.00
Bridge04	97.97	95.54	98.54	96.97	94.16
Bridge05	96.75	94.31	94.45	94.32	89.53
Bridge06	98.06	93.60	98.55	95.84	92.26
Bridge07	98.77	97.23	99.02	98.11	96.30
Bridge08	96.79	95.70	97.40	96.47	93.22
Bridge09	95.92	94.66	92.62	93.45	88.06
Bridge10	92.83	94.26	92.82	93.24	87.49
Avg.	96.97	94.59	96.63	95.48	91.57

The class-wise comparison of the models can be seen in Table 7.4, where the evaluation metrics across classes and their average values obtained from the LOOCV test have been shown. As can be seen, the Railing class has been the most challenging for both models, yielding lower accuracy values compared to other classes.

Table 7.4: Class-wise comparison between MSFD-Net and RandLA-Net.

Metric	Model	Background	Deck	Abutment	Railing	Avg.
Precision	MSFD-Net	97.39	97.20	95.80	87.97	94.59
	RandLA-Net	97.93	95.63	92.91	76.33	90.70
Recall	MSFD-Net	97.56	95.51	99.28	94.20	96.63
	RandLA-Net	95.67	96.32	99.37	90.95	95.58
F1	MSFD-Net	97.43	96.22	97.50	90.75	95.48
	RandLA-Net	96.76	95.92	96.02	82.33	92.76
IoU	MSFD-Net	95.02	92.84	95.14	83.28	91.57
	RandLA-Net	93.75	92.18	92.38	70.50	87.20

One of the primary reasons contributing to the difficulty in predicting this class lies in its limited representation within the dataset. This scarcity of data points allocated to this class diminishes the models' ability to effectively learn its patterns and characteristics. Even with the implementation of class weights, intended to mitigate the imbalanced class issue, the model still cannot find adequate samples to learn. When models encounter fewer samples of a particular class, their performance is restricted. Furthermore, the situation escalates when models opt to subsample points to alleviate the processing load. Subsampling, while aiding computational efficiency, further limits the exposure to classes such as Railing, compounding the challenge of pattern recognition and accurate prediction within this class.

Considering the class-wise results of the models, MSFD-Net has been able to significantly improve this class's prediction results. This implies that this model can learn more efficiently even with lower number of point samples. The results also illustrate that MSFD-Net has outperformed RandLA-Net in predicting the point label of most classes. In terms of IoU and F1 scores, MSFD-Net has shown a superior performance for all classes. It has also achieved a more promising performance for most classes in terms of Precision and Recall. Further discussion and details about the models can be found in Appendix A.

The visual comparison in Figure 7.3 illustrates the performance difference between RandLA-Net and MSFD-Net across three typical bridge samples throughout the LOOCV test. As can be seen, MSFD-Net demonstrates a higher accuracy in classifying points. In Bridge 01, RandLA-Net misclassifies points belonging to the Background class as the class Deck.

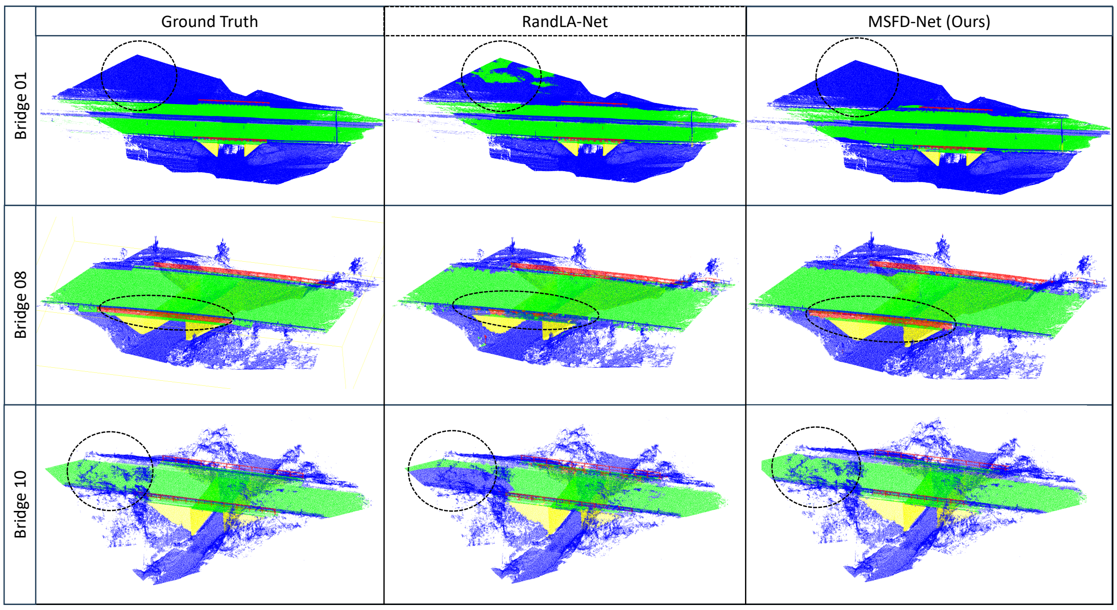


Figure 7.3: Visual comparison of semantic segmentation results for three bridge samples. Class colors: Blue: Background, Green: Deck, Red: Railing, and Yellow: Abutment.

Furthermore, in Bridge 08, it fails to detect one of the railing instances completely. The challenges persist in Bridge 10, where the model encounters difficulty in accurately classifying points associated with the Deck class. As can be seen, MSFD-Net has been capable of alleviating classification errors in these samples and providing more reliable results for the digital twinning of bridges.

7.3 Preparation of Segmented Point Clouds

Bridge point clouds need to be prepared prior to applying PPMs and deriving the value of parameters. Figure 7.4 depicts the required steps, including semantic segmentation, transformation, instance segmentation (clustering), and face/cross-section detection. All these steps must be followed before starting the parametric modeling process (second major block).

Considering a raw bridge point cloud, semantic segmentation is the initial step in processing points, as shown in Figure 7.4a. In this research, MSFD-Net has been proposed for semantic segmentation of bridge point clouds (Chapter 4) as it shows a reliable performance in classifying points and reaches a level of accuracy whose results can be properly used by the downstream modules/algorithms. However, the parametric modeling module is not limited to this model only. Therefore, other existing methods for semantic segmentation of point clouds, including bottom-up [134, 231, 232], top-down [143, 145, 146], or other deep learning models [163, 165, 155] can be employed as well.

Semantic segmentation separates the raw bridge point cloud into the point cloud of bridge elements. It also narrows down the initial problem from the entire bridge point cloud to the point cloud of bridge elements and determines the type of each component from which the PPM can be recognized. As described in the previous section, four classes have been considered to classify points: Background, Deck, Abutment, and Railing. As the points belonging to the class Background are not of interest for the geometric modeling bridges, they are deleted, and the problem is simplified to the three remaining classes for digital twinning.

The raw bridge point clouds are not generally along the x-axis and have some degrees of rotation around the z-axis. For bridge point clouds with a straight deck (without a large horizontal curvature), it is more suitable to rotate the point cloud around the z-axis and make it along the x-axis. Thus, transformation (translation and rotation) of the segmented point clouds is the next preprocessing step, as shown in Figure 7.4b. As the variance of points along the length of the bridge deck is significantly higher than in the other directions, principal component analysis (PCA) is employed to detect the

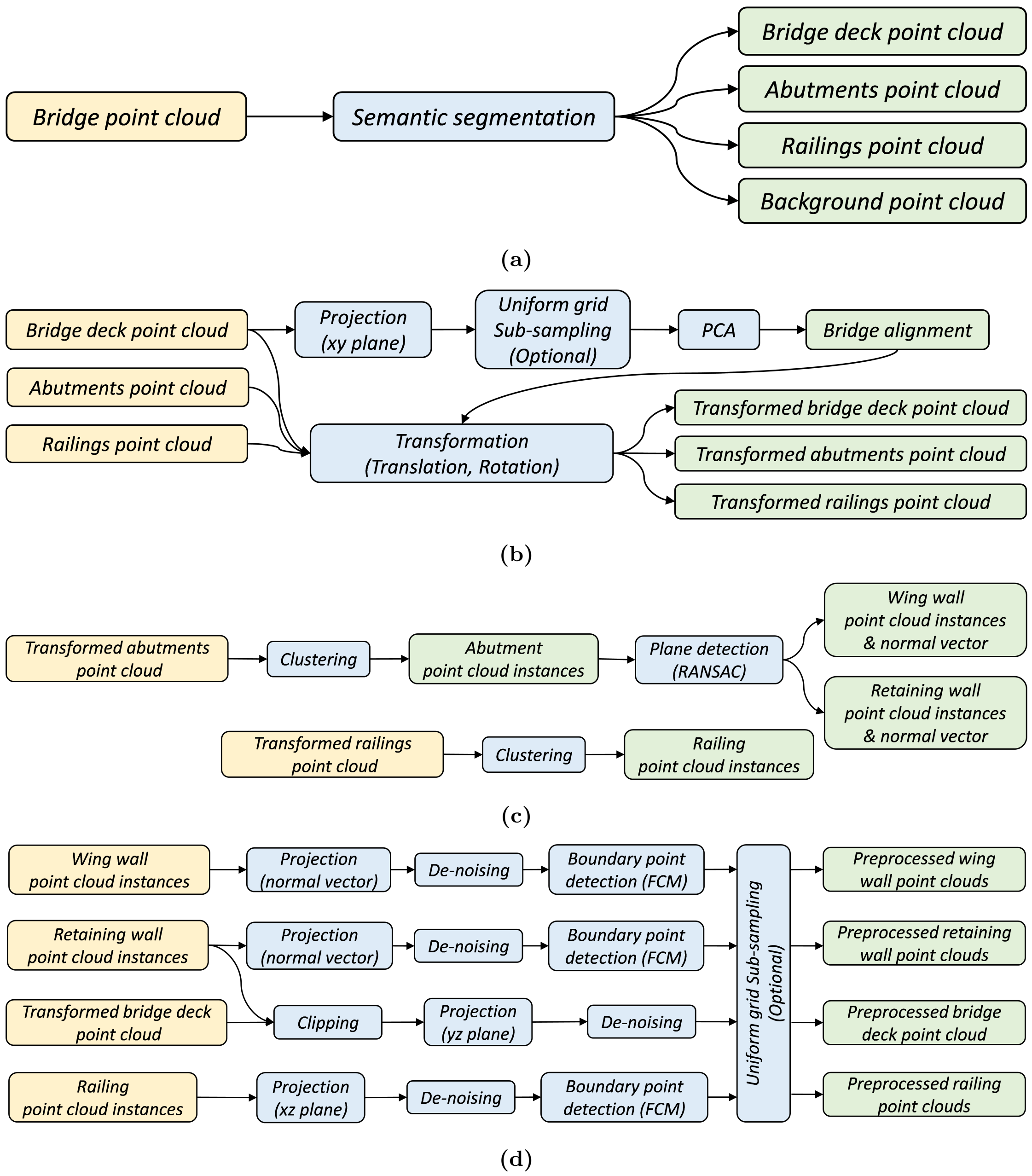

Bridge point cloud
Semantic segmentation
Bridge deck point cloud
Abutments point cloud
Railings point cloud
Background point cloud
(a)
Bridge deck point cloud
Abutments point cloud
Railings point cloud
Projection (xy plane)
Uniform grid Sub-sampling (Optional)
PCA
Bridge alignment
Transformation (Translation, Rotation)
Transformed bridge deck point cloud
Transformed abutments point cloud
Transformed railings point cloud
(b)
Transformed abutments point cloud
Clustering
Abutment point cloud instances
Plane detection (RANSAC)
Wing wall point cloud instances & normal vector
Retaining wall point cloud instances & normal vector
Transformed railings point cloud
Clustering
Railing point cloud instances
(c)
Wing wall point cloud instances
Projection (normal vector)
De-noising
Boundary point detection (FCM)
Retaining wall point cloud instances
Projection (normal vector)
De-noising
Boundary point detection (FCM)
Transformed bridge deck point cloud
Clipping
Projection (yz plane)
De-noising
Railing point cloud instances
Projection (xz plane)
De-noising
Boundary point detection (FCM)
Uniform grid Sub-sampling (Optional)
Preprocessed wing wall point clouds
Preprocessed retaining wall point clouds
Preprocessed bridge deck point cloud
Preprocessed railing point clouds
(d)

Figure 7.4: Required preprocessing steps for the proposed method: (a) semantic segmentation; (b) transformation; (c) clustering (instance segmentation); (d) cross-section/face and boundary points detection.

alignment of the bridge. To this end, the point cloud of the bridge deck is projected onto the xy plane, and a uniform grid subsampling is applied to remove the impact of overlying points resulting from the projection. Then, PCA is executed, and the segmented point clouds are translated and rotated around the z-axis as much as the angle between the principal component obtained by PCA and the x-axis.

There is generally more than one point cloud instance in the classes of abutments and railings. Also, abutments consist of sub-elements, including a retaining wall and two wing walls. Therefore, these classes need to be further segmented/clustered, as shown in Figure 7.4c. To this end, the clustering algorithm described in Section 5.1 is employed to detect the two instances in each class. As mentioned, this algorithm clusters the point cloud instances following the connectivity rules. As the point cloud instances, such as abutments and railings, generally stand far from each other, a connectivity radius of $r = 1$ m is considered for the instance segmentation. In order to detect the point cloud of the retaining wall and the wing walls, the RANSAC algorithm is employed. As the number of existing faces in each abutment point cloud is known (two wing walls and a retaining wall), the number of existing thresholds in RANSAC is reduced and limited to only a distance threshold from the planes that can be reasonably selected (herein 10 cm), over a number of iterations (herein 300) for each plane instance.

The last remaining step is the detection of cross-section or boundary points of faces, which are required for fitting PPMs. For this purpose, a combination of projection, de-noising, FCM clustering, and subsampling functions/methods is employed, as shown in Figure 7.4d. The point cloud of wing walls and retaining walls is projected onto 2D planes using their normal vectors detected by the RANSAC algorithm in the previous step. The boundary points are then detected by the FCM clustering algorithm proposed in Section 5.2. As the bridge deck point cloud is the part between the retaining walls, it is clipped and projected onto the yz plane. The railing point clouds are also projected onto the xz plane, and their boundary points are detected using the FCM clustering algorithm. All these point clouds are de-noised after projection, as described in Section 5.1, and subsampled by uniform grid subsampling (grid size $\simeq$ 5 to 20 cm). In all the steps, the subsampling module is optional and can be eliminated. This module has been only used to decrease the processing loads of the algorithms and remove the impact of overlying points due to the projection. The de-noising module also checks the connectivity rules to ensure no point exists far from the target points.

7.4 Parametric Modeling

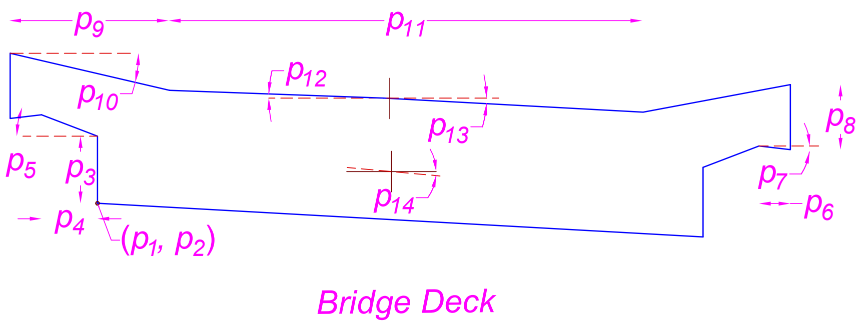
This section evaluates the performance of the proposed approach in creating the geometric DT of bridges. It consists of two real-world experiments. First, the entire method is applied to the point cloud of ten single-span RC bridges, and the geometric DT of the entire bridge is created. Second, PPMs are applied to more complicated geometric shapes, such as T-shaped and hollow bridge decks and piers, that mostly exist in multi-span bridge point clouds to show the wide range of applications the algorithm can have. Next, the proposed approach is compared with three other approaches in the literature and its advantages are highlighted. Finally, PPMs are applied to point clouds with large occasions, and their performance is compared with other conventional modeling algorithms.

7.4.1 Experiment 1: Geometric Modeling of Ten Single-Span RC Bridges

To derive the value of parameters, the corresponding PPM to each preprocessed point cluster is selected. As shown in Figure 7.4, the semantic segmentation and the clustering modules generally determine the type of the required PPMs for model fitting. However, in scenarios where the type of PPMs is not known, the supervised and unsupervised selection methods (Section 6.4) can be employed.

Figure 7.5 shows the details of PPMs used for model fitting all the bridge samples. These PPMs include a bridge deck, wing wall, retaining wall, and railing obtained by analyzing the bridge point cloud samples to reach a desired LoD. All the PPMs have been initialized only once and used for the geometric digital twinning of all the bridge samples, i.e., no user intervention is applied to the PPMs from sample to sample. Most of the parameter intervals have been obtained by analyzing a large number of bridge data provided by the German bridge database "SIB-Bauwerke" as well as empirical knowledge. For parameters such as the origin or width of the bridge deck that might largely vary in bridges, the axis-aligned bounding box (AABB) of the point clouds, with the lower left corner (ll) and upper right corner (ur), has been used to relatively set the initial values. All these intervals have also been shown in Figure 7.5.

To adjust the instantiated PPMs and fit them into their corresponding point clouds, TLBO is employed as it showed promising performance in Section 6.5. This algorithm only needs a number of population/particles (75 particles) and a stopping criteria (300 iterations). Piece-wise optimization problems are solved by TLBO for each point cluster representing a bridge element. Each optimization process starts with a list of candidates randomly generated by the optimization algorithm. This list is further refined by TLBO such that the PPMs can be fitted into the point cloud. This process leads to a close approximation of the parameter values after optimization. Considering the number of four wing walls, two retaining walls, two railings, and a bridge deck that exists in each bridge point cloud, the optimization algorithm must be capable of extracting the value of 52 parameters. To evaluate the accuracy of the resulting models, the mean absolute error (MAE) of PPMs is calculated by Equation 6.1 that shows the distance of points to PPMs.



p_1: x-origin $\in [x_{ll}, x_{ur}]$

p_2: y-origin $\in [y_{ll}, y_{ur}]$

p_3: deck depth $\in [0.05, 1.30]$

p_4: cantilever width $\in [0.10, 1.80]$

p_5: cantilever slope $\in [5°, 75°]$

p_6: parapet bottom width $\in [0.05, 1.00]$

p_7: parapet bottom slope $\in [-45°, 0°]$

p_8: parapet height $\in [0.05, 1.20]$

p_9: parapet top width $\in [0.50, 3.50]$

p_{10}: parapet top slope $\in [-45°, 0°]$

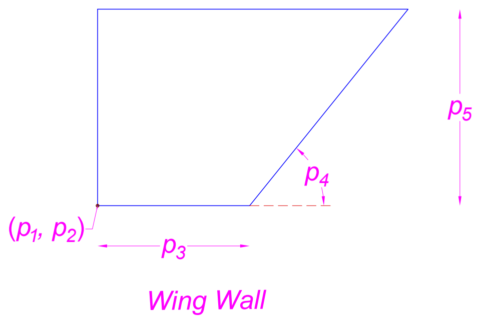
p_{11}: deck width $\in [4.00, x_{ur} - x_{ll}]$

p_{12}: deck right slope $\in [-5°, 5°]$

p_{13}: deck left slope $\in [-5°, 5°]$

p_{14}: deck inclination $\in [-10°, 10°]$

(a)



p_1: x-origin $\in [x_{ll}, x_{ur}]$

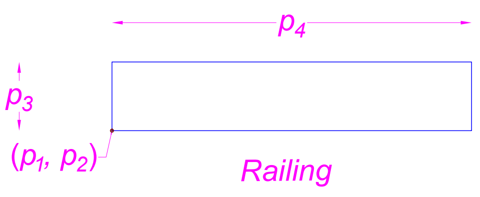
p_2: y-origin $\in [y_{ll}, y_{ur}]$

p_3: width $\in [0.30, x_{ur} - x_{ll}]$

p_4: slope $\in [30°, 80°]$

p_5: height $\in [2.00, y_{ur} - y_{ll}]$

(b)



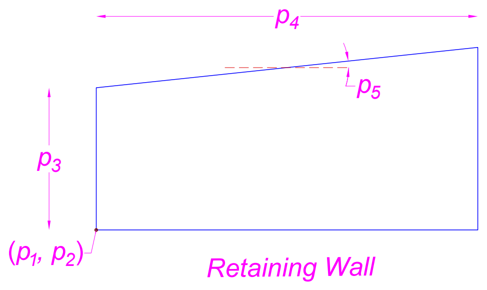
p_1: x-origin $\in [x_{ll}, x_{ur}]$

p_2: y-origin $\in [y_{ll}, y_{ur}]$

p_3: height $\in [0.50, 1.80]$

p_4: length $\in [0.80 \times (x_{ur} - x_{ll}), x_{ur} - x_{ll}]$

(c)



p_1: x-origin $\in [x_{ll}, x_{ur}]$

p_2: y-origin $\in [y_{ll}, y_{ur}]$

p_3: height $\in [2.00, y_{ur} - y_{ll}]$

p_4: length
$\in [0.80 \times (x_{ur} - x_{ll}), x_{ur} - x_{ll}]$

p_5: slope $\in [-10°, 10°]$

(d)

Figure 7.5: List of PPMs used for geometric digital twinning of the bridge point clouds: (a) bridge deck; (b) wing wall; (c) railing; (d) retaining wall. x_{ur}, y_{ur} and x_{ll}, y_{ll} are the x- and $y-$ coordinate of the upper right and lower left corner of the axis-aligned bounding box (AABB) surrounding the input point cloud. $x_{ur} - x_{ll}$ and $y_{ur} - y_{ll}$ are also the length and the height of the AABB, respectively. The dimensions of the AABB are used for the relative initialization of PPMs. All values are in meter (m).

Table 7.5 illustrates the MAE of PPMs after the model fitting process. Averaging the resulting values of error from the bridge samples shows that TLBO has been capable of modeling bridges with an MAE of 8.43 cm. Note that noises and other imperfections in the entire method have been considered in the calculation of MAE. Therefore, these error values show the worst case in which some noises or wrongly classified points still exist in the problem space. In addition, no external intervention has been made in the modeling process of bridges from the point clouds. This table also demonstrates a class-wise comparison of each element's error value. As can be seen, the class Retaining Wall (RW) has resulted in the highest value of MAE.

Table 7.5: MAE of the fitted PPMs into the point cluster of bridge elements (cm).

Sample	Wing Wall (WW)				Retaining Wall (RW)		Railing (R)		Deck	Mean
	WW1	WW2	WW3	WW4	RW1	RW2	R1	R2		
Bridge 01	1.87	2.65	2.26	2.45	11.73	9.76	6.00	5.65	13.00	6.15
Bridge 02	6.67	5.48	8.08	5.75	35.26	37.06	7.89	9.91	12.97	14.34
Bridge 03	6.30	6.71	6.63	6.19	12.82	15.19	15.71	17.86	7.61	10.56
Bridge 04	3.28	3.28	4.83	4.83	4.74	9.65	17.84	17.85	8.05	8.26
Bridge 05	2.94	2.94	3.15	2.88	7.03	7.54	16.09	7.99	15.51	7.34
Bridge 06	2.82	2.37	2.39	3.07	9.05	4.25	10.32	9.95	5.98	5.58
Bridge 07	3.34	2.91	4.33	3.12	7.37	7.54	1.66	1.13	3.52	3.88
Bridge 08	4.62	3.84	17.73	5.42	20.16	18.64	13.04	9.25	4.15	10.76
Bridge 09	15.54	11.99	22.58	22.58	7.09	7.71	2.19	2.55	5.09	10.81
Bridge 10	4.93	6.10	7.76	6.45	6.22	7.73	7.86	6.36	5.95	6.60
Mean	6.08				12.33		9.35		8.18	8.43

Figure 7.6 shows the fitted model to the retaining wall of Bridge 02 after optimization. As can be seen, the bottom and vertical edges of the PPM have horizontal and vertical constraints, thus preventing them from rotating and becoming closer to the points. This instance shows that the governing reverse engineering approach and the injected bridge engineering knowledge enforce the algorithm to generate PPMs that necessarily end up with the anticipated 3D model. Although the rotational degrees of freedom can be given to such edges, the 3D model must also be capable of accepting these new parameters as the process has been started from the final model. In this example, our presumption has been to generate a bridge model whose retaining walls have constraints on the bottom and lateral edges.

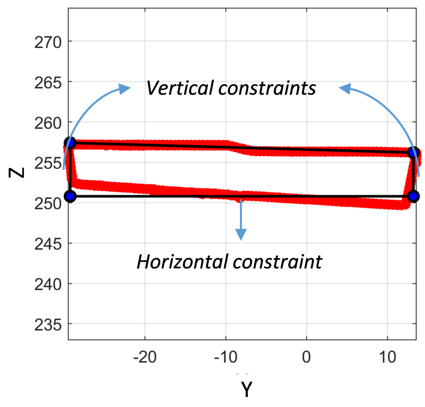


Figure 7.6: Fitted retaining wall of Bridge 02 by the PPM.

To generate the 3D model of the entire bridge, all the extracted parameters have been assembled by solving another optimization problem as described in Section 6.3. In this process, all the involved PPMs are refined again so that the common parameters among elements are integrated, and the PPMs still remain as close as possible to their corresponding point cloud.

Figure 7.7 depicts the histogram of each bridge sample after the geometric modeling process. The vertical axis of the diagram shows the number of points assigned to all the involved PPMs, and the horizontal axis shows the distance of points to the PPMs in terms of MAE. As can be seen, a large portion of points has a distance of less than 5 cm from the fitted PPMs in all the samples. However, in sample Bridge 02 (Figure 7.7b), the variation range of MAE is larger than a sample such as Bridge 07 (Figure 7.7g). This observation is also compatible with Table 7.5 in which the value of MAE is higher in Bridge 02. Comparing the point cloud of this sample with other samples shows that Bridge 02

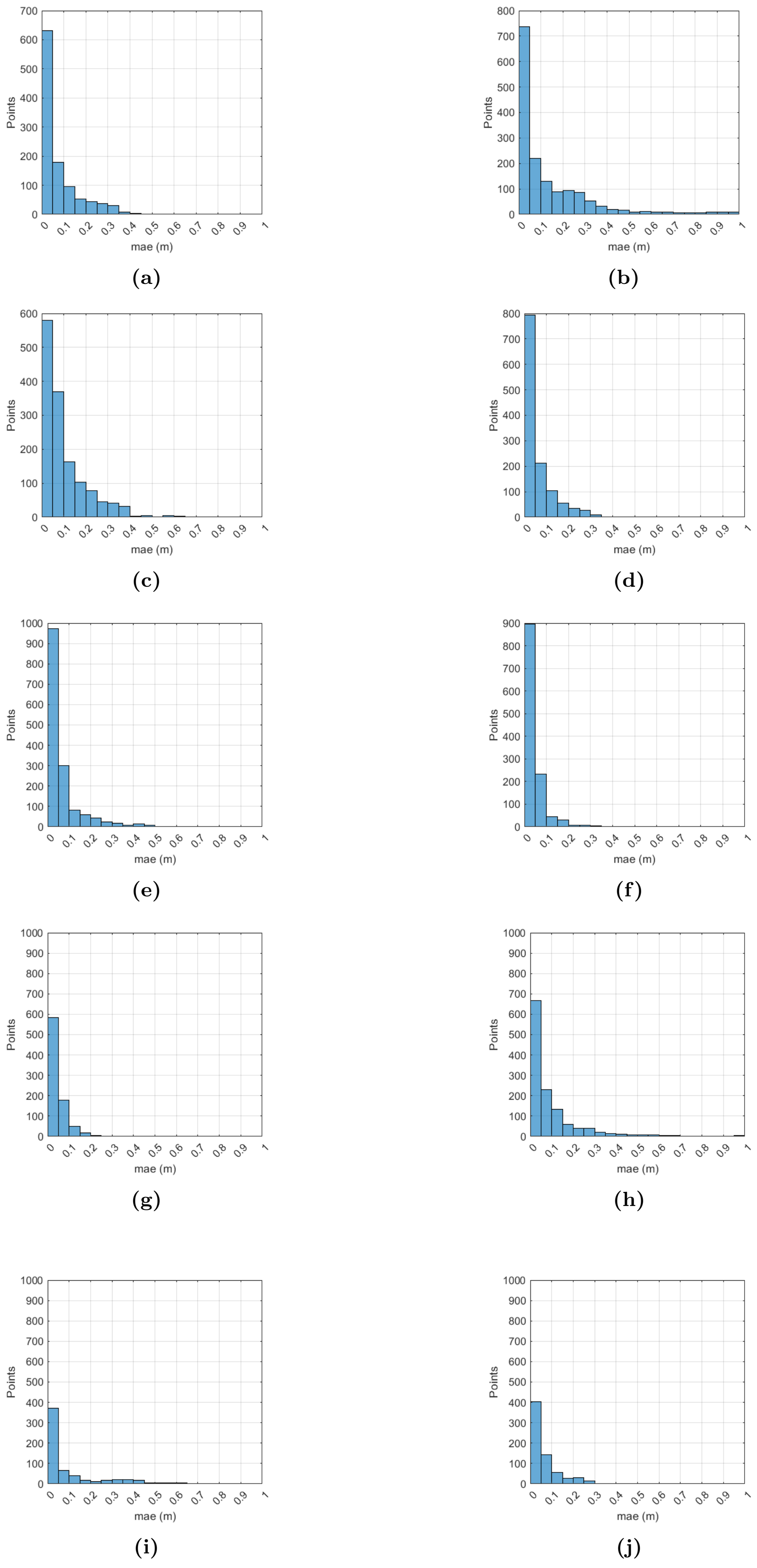


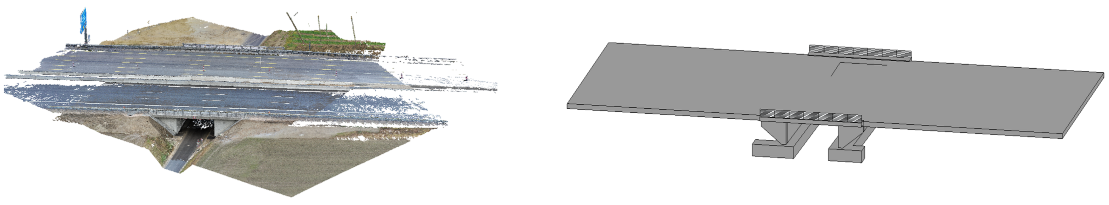
Figure 7.7: Histogram of bridge samples after model fitting: (a-j) show the bridge samples 01-10, respectively.

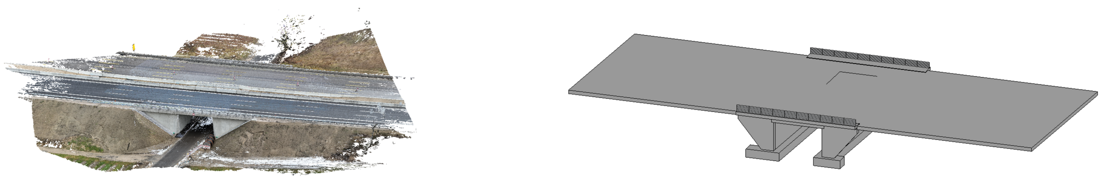
has more differences in type with respect to the desired model. Hence, the generated PPMs from the model at the beginning of the process have not been able to completely describe/capture differences beyond the imposed presumptions/restrictions. This means the four PPMs used for modeling all the bridge components of the ten samples have been more compatible in type (not dimensions; the same setup/initialization has been used for all the samples) with the point cloud of other samples. As a result, in Bridge 02, the edges and vertices of the PPMs have not been able to move closer to the bridge point cloud, which, in turn, has led to a higher value of error. Table 7.6 shows the overall time required for preprocessing segmented point clouds, extracting the value of parameters, and assembling them into an integrated model. As can be seen, the modeling time of all the samples with around 2 million points is less than 370 sec (6.16 min).

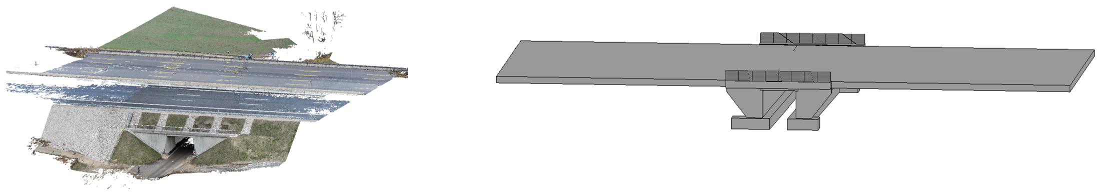
Table 7.6: Required time for modeling bridges from point clouds.

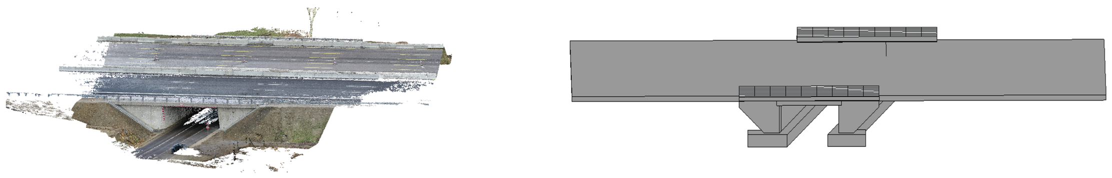
Sample	Bridge 01	Bridge 02	Bridge 03	Bridge 04	Bridge 05	Bridge 06	Bridge 07	Bridge 08	Bridge 09	Bridge 10
Time (sec)	285.41	367.07	328.16	311.76	350.18	305.01	192.07	239.94	233.03	161.15

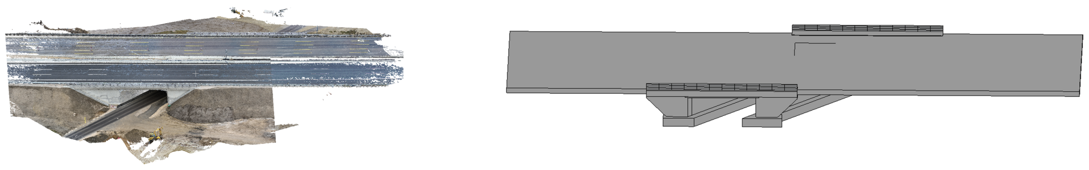
This shows the massive reduction of the modeling time in comparison with the manual modeling processes, which usually take several days. To visualize the 3D model of each bridge sample, the parameter values are imported into the 3D PPM of the bridge. This process leads to the 3D geometric model of each bridge sample as shown in Figure 7.8.

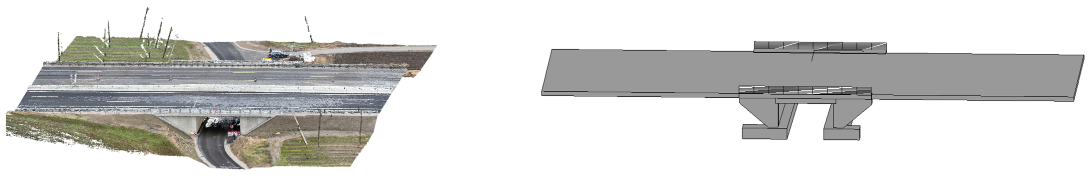

(a)

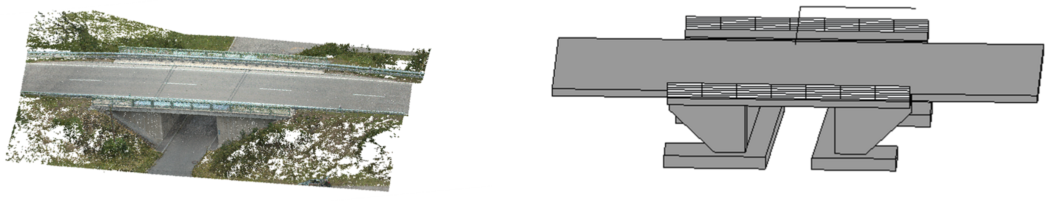

(b)

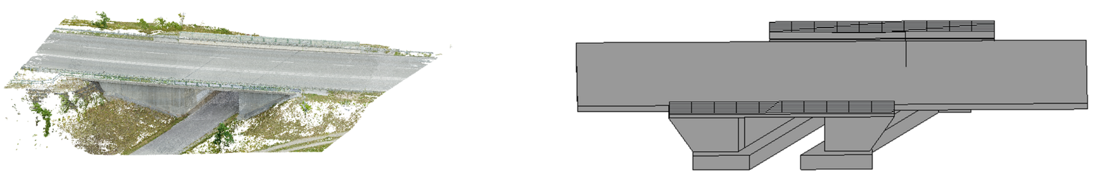

(c)


(d)

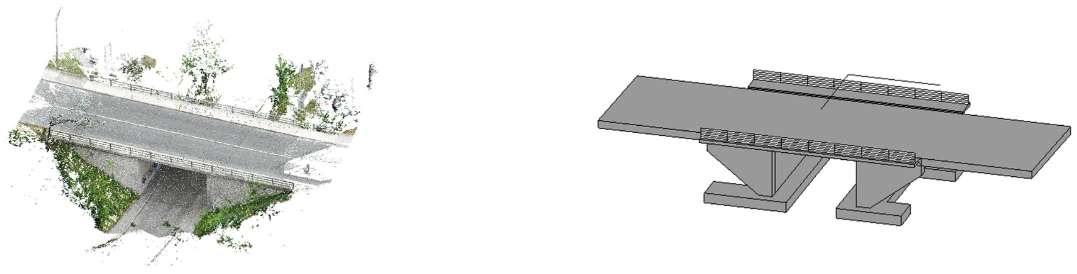

(e)


(f)

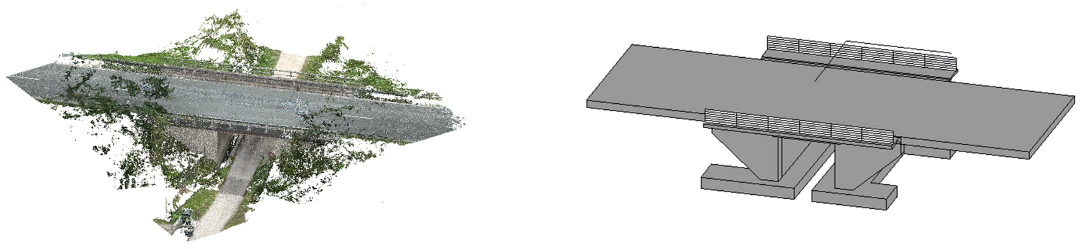

(g)



(h)



(i)



(j)

Figure 7.8: Point cloud of bridges and their corresponding geometric DT model. (a-j) show the bridge samples 01-10, respectively.

7.4.2 Experiment 2: Bridge Elements

Contrary to single-span bridges, multi-span bridges have a longer deck supported by piers. The deck of multi-span bridges generally has vertical and horizontal curvatures and cannot be described properly by a single extrude function.

Figure 7.9 shows the process of modeling the deck of a typical multi-span bridge. To capture the curvature and changes of such bridges, the alignment of the deck needs to be detected in the initial step. The alignment of straight bridges/decks can be recognized using the PCA algorithm similar to Experiment 1. In the case of bridges with horizontal curvature, a polynomial can be fitted to the deck point cloud after projection onto the

xy plane. Using the bridge alignment, the deck point cloud can be split into smaller segments, each of which is placed between two planes/sections in a pre-defined distance (Δ) (see Figure 7.9). These segmented point clouds can be projected onto a 2D plane and fitted using a PPM by solving multiple piece-wise optimization problems. Note that a single PPM with the same initialization is used for fitting all the slices of the bridge deck point cloud. This model-fitting process leads to a list of parameters obtained from each slice. Sweeping/connecting all the PPMs along the length of the bridge deck results in the 3D model of the deck. However, this model might not be smoothed as the extracted values from PPMs have differences along the length of the bridge deck (Figure 7.9). To address this issue, the values of each parameter are regularized separately in three steps. First, assuming a normal distribution for parameter values, the outliers are removed by calculating the mean value (μ) of the parameter and its standard deviation (σ). Second, a polynomial is fitted to the values. Third, the value of the parameter is read from the fitted polynomial using its location.

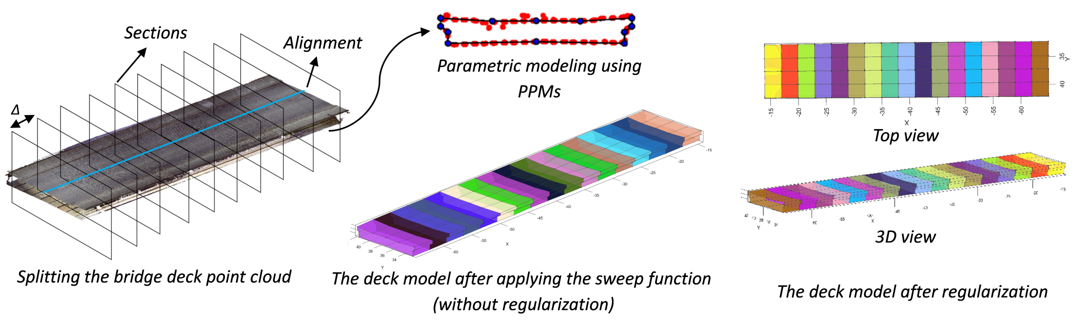
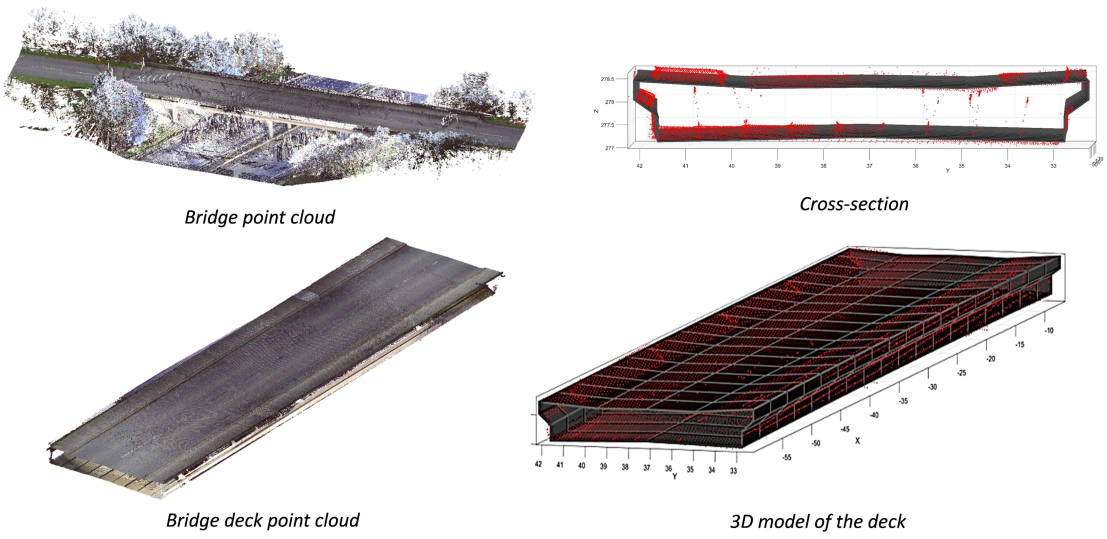

Sections
Alignment
Δ
Splitting the bridge deck point cloud
Parametric modeling using PPMs
The deck model after applying the sweep function (without regularization)
Top view
3D view
The deck model after regularization

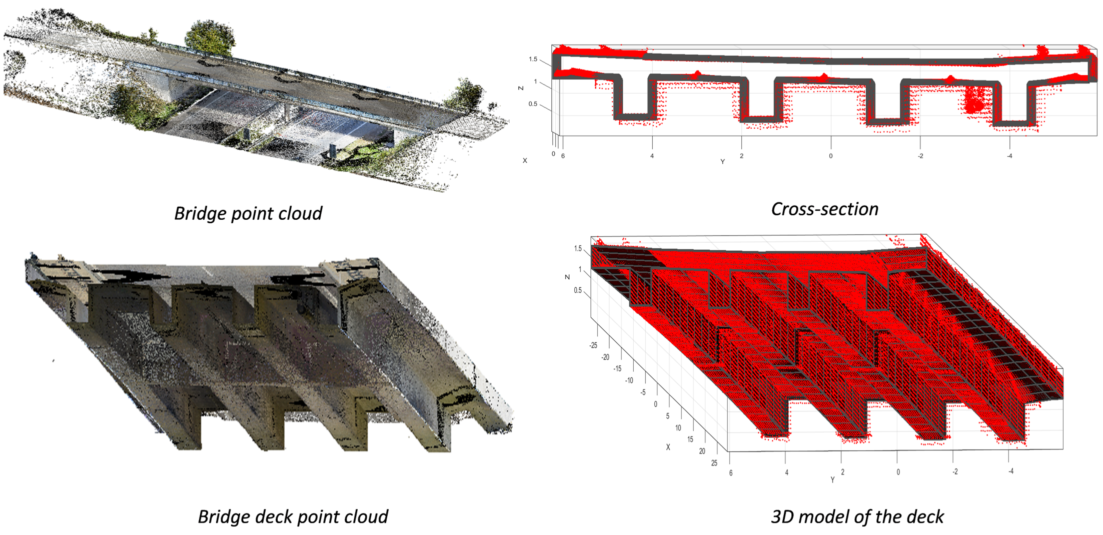
Figure 7.9: Modeling process of a typical multi-span bridge deck.

To clarify, assume a deck point cloud with a starting point at $x = 0$ and an endpoint at $x = 3$. Considering four sections at locations such as $x = \{0, 1, 2, 3\}$, three segments of the point cloud can be obtained and fitted by a PPM. Let $p = \{2.40, 2.50, 2.60\}$ be the extracted values for a parameter such as the parapet width by fitting the three sequential PPMs. After removing outliers from the set p, for ex., values more and less than $\mu \pm \sigma$ (68% of data), a polynomial, such as $ax^2 + bx + c$, can be fitted to the values of the set p. Using this polynomial whose coefficients are known after fitting, regularized values of the parameter can be extracted by inserting the values of the set x into the fitted polynomial (Figure 7.9).

Figure 7.10a shows the results of applying PPMs in the parametric modeling of two multi-span bridges. The first bridge sample is Bridge 01 from the Cambridge bridge point cloud dataset [143], which shows a concrete bridge point cloud acquired by laser



(a)



(b)

Figure 7.10: Parametric modeling of two bridge decks from their point clouds: (a) first sample; (b) second sample.

scanning. As the bridge deck is straight, PCA can be applied to this sample similarly to the single-span bridges. To generate the geometric model, the bridge deck is split into intervals of 2 m, and a PPM is fitted to each segment of the point cloud. The PPM of this sample is similar to the PPM used for the deck of single-span bridges. After extracting the value of parameters, outliers are removed, and a polynomial of degree two is fitted. As can be seen in Figure 7.10a, the model has been fitted into the point cloud completely.

Calculating the distance of points to the PPMs along the length of the bridge deck shows an MAE of 1.67 cm/m, while noises have also been considered in calculating the value of error; thus, the computed value shows the worst case. Figure 7.10b also demonstrates the application of PPMs in the geometric modeling of another multi-span bridge captured in Munich, Germany. Contrary to Experiment 1, this sample has been acquired through laser scanning. In comparison with the previous bridge sample, this bridge deck is more complicated in shape as it has four T-shaped concrete girders. To select the appropriate type of PPM for describing the point cloud sample, the unsupervised selection method proposed in Section 6.4 has been used, and T-shaped bridge decks with 3-6 girders are tested. As the value of MAE resulting from PPM with four girders has been lower, this type of PPM is selected.

This PPM can also be initialized similarly to the deck of single-span bridges. The only difference is the existence of girders whose dimensions can be logically set up based on bridge engineering knowledge. In this example, a width of [0.1, 1.5] and a depth of [0.2, 1.5] have been considered for the girders. Also, they have a distance of [0.5, 3] with respect to each other, half of which belongs to the flange of a girder. The flange can also have a value of slope in the range of [-3°, 3°]. This bridge deck point cloud has a length of around 80 m, and it has been sliced every 2.5 m. This means $80/2.5 = 32$ distinct optimization problems that need to be solved to model this bridge deck. As can be seen in Figure 7.10b, a single PPM has been capable of deriving the parameter values from all the slices and generating the 3D model of the deck. Averaging the values of MAE over the length of this bridge shows a value of 1.05 cm/m while noises still exist in the problem.

Various types of piers can be seen in bridges. A common type of bridge pier is shown in Figure 7.11, which consists of a pier cap and two pier columns. This pier can be modeled by PPMs if the pier cap is separated from the pier column. To this end, an FCM clustering algorithm is used for two clusters (pier cap and pier column). As the feature vector of the FCM, three features are calculated that represent differences between these two elements. First, the pier cap is generally over the pier columns; thus, its points have higher values of z-coordinate. Second, the pier cap is a horizontal element while the pier column is vertical; therefore, the z-component of the points' normal vector is higher for the pier cap. Third, if the pier is projected onto the xy plane, the 2D density of the points belonging to the pier column is higher as it is a vertical element. The 2D density can be calculated by counting the number of neighboring points placed within a circle with a predefined radius. Using these three features, the point cloud can be segmented.

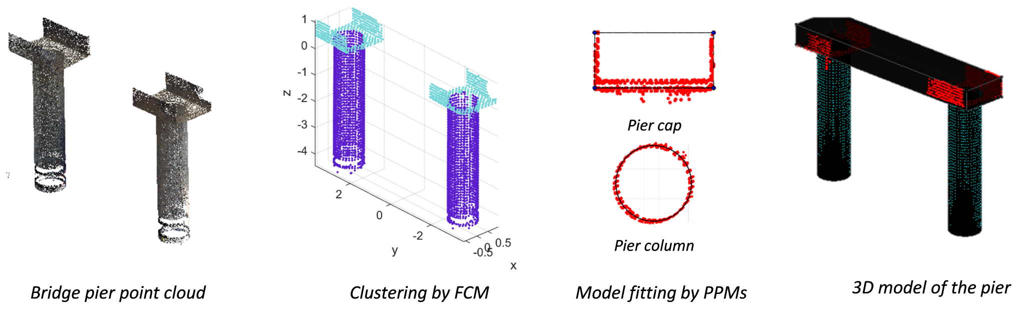


Figure 7.11: Modeling process of a typical bridge pier.

To extract the value of parameters from the pier column, the points of each pier are projected, and a circular PPM is fitted. Note that a circle is a primitive shape; however, it can still be represented with three parameters, a center (x, y) and a radius (r), and its distance to the points is minimized by a metaheuristic algorithm. The pier cap can also be modeled by a rectangular PPM. As can be seen in Figure 7.11, the pier cap has occlusion and some noise; however, the PPM can still perform properly. After extracting the value of parameters, the elements can be assembled, and the 3D model of the pier is obtained. Averaging the value of MAE from fitting the pier columns and the pier cap shows an MAE of 1.43 cm.

7.4.3 Discussion

The performance of the proposed method can be evaluated in various scenarios and compared with other existing algorithms. This section further discusses the model fitting process by PPMs and highlights its advantages in geometric modeling.

7.4.3.1 Comparison with Other Methods

For comparison, the point clouds of an I-shaped beam and a bridge deck are fitted by PPMs, α-Concave hull [233], and RANSAC algorithm [233]. Figure 7.12 visually compares these methods after applying each algorithm. As can be seen, PPMs have been more successful in model-fitting, thanks to the reverse engineering strategy governing the optimization algorithm. The other two methods cannot provide an exact number of parameters after model fitting and require another heuristic algorithm to refine their results. Therefore, these methods cannot directly provide a meaningful parametric model without any post-processing step. The proposed algorithm, however, results in a finite number of parameters with a close approximation of their values. It also preserves constraints such as orthogonality, parallelism, and symmetry in model fitting to meet the anticipated requirements.

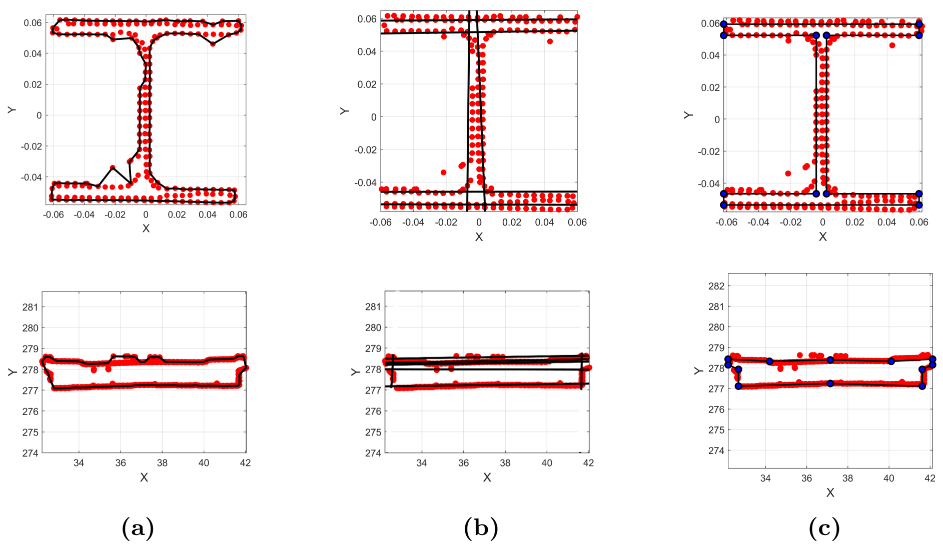


Figure 7.12: Comparison the results of the model fitting approaches: (a) α-Concave hull [233]; (b) RANSAC [119]; (c) PPM (ours).

Table 7.7 compares the proposed method with the most recent methods [132, 210, 201] in the geometric modeling of bridges or structural elements. Each table column represents a feature that can be a basis for comparison. The second column demonstrates the bridge components addressed by the methods. As can be seen, the two first methods are limited in covering all the components that generally exist in bridges and have mainly focused on steel profiles/sections, while the third method, in addition to steel girders, covers piers and the bridge deck. The third column in the table shows the core model-fitting algorithm used to model the geometry from point clouds. The first method utilizes the RANSAC algorithm for estimating the dimensions of steel profiles, while the second method uses a kernel density estimation (KDE) algorithm to detect the type of cross-section from a catalog. The third method also employs α-concave hull for more complicated geometries, such as the bridge deck, and a density estimation algorithm to detect the type of girders from a catalog. The column named "Modeling Level" shows the coverage level of the proposed approaches. The first two methods have been limited to modeling bridge components, while the next two methods have generated the entire model of the bridge. The Assembly column demonstrates whether the assembly process of elements has been described or not. As can be seen, none of the other methods have addressed this problem.

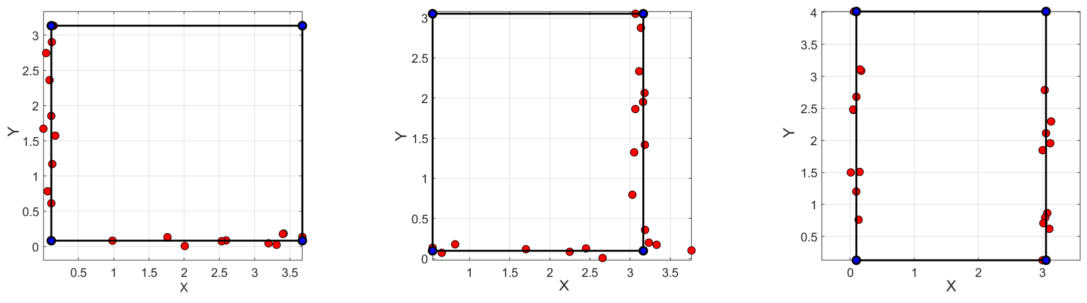
Table 7.7: Comparison of the proposed method with three state-of-the-art methods.

Method	Covered Elements	Core Algorithm	Modeling Level	Assembly	Accessibiliy to Dimensions	Parametric Modeling
1. Yan and Hajjar [132]	super-structure components: I-shaped girders and cross-frames	RANSAC	Components	No	Yes (only steel sections)	No
2. Laefer and Truong-Hong [210]	Steel sections: I-, L-, T- and C-shape sections	KDE	Components	No	Yes (only steel sections)	No
3. Lu and Brilakis [201]	super- and sub-structure components: bridge girders, piers, and decks	α-concave hull & density estimation	Entire bridge	No	Yes (only circular pier columns and steel sections)	No
4. Ours	super- and sub-structure components: retaining walls, wing walls, parapets, bridge girders, piers, railings, and decks	PPM	Entire bridge	Yes	Yes (all elements)	Yes

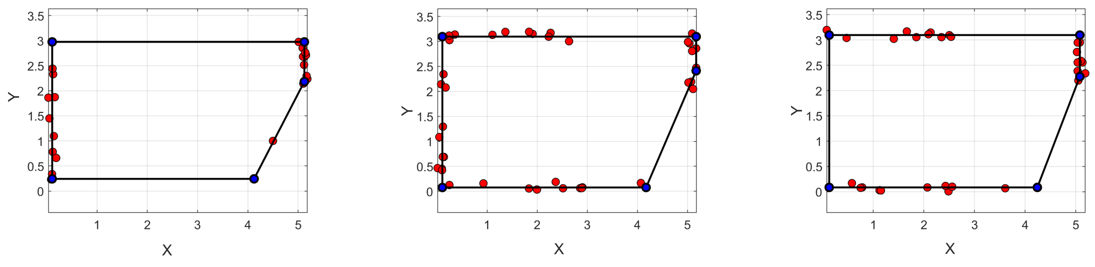
The next column (Accessibility to Dimensions) represents whether the value of parameters/dimensions has been extracted from point clouds. The first two methods have been capable of obtaining the value of parameters. However, these methods have only covered steel sections such as girders or cross-frames. The third method has also been limited and only extracted the value of parameters for circular pier columns and steel girders. This method uses the α-concave hull for describing the more complicated geometries, and as discussed in Figure 7.12, this algorithm cannot solely result in the parameter values. The last column also shows whether the resulting model is parametric and can accept geometric updates. As can be seen, none of the other methods have included this feature in the geometric modeling.

7.4.3.2 Occlusion Resistant Model-Fitting

The proposed concept of *active* and *passive* edges can improve the algorithm's performance in fitting PPMs into occluded point clouds. This new definition of fitness function can generate results at a competitive level with human recognition in modeling. Figure 7.13 shows the results of the model fitting a rectangular and a trapezoidal PPM into the occluded point clouds.



(a)



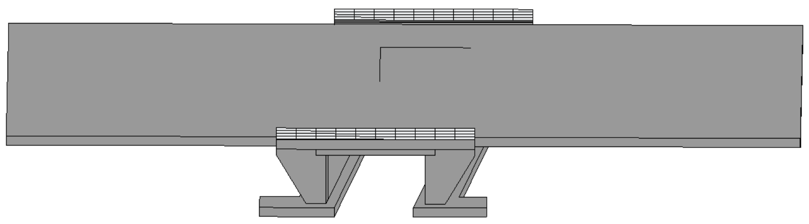
(b)

Figure 7.13: Performance of the algorithm in sustaining a large amount of occlusion: (a) a rectangular PPM; (b) a trapezoidal PPM.

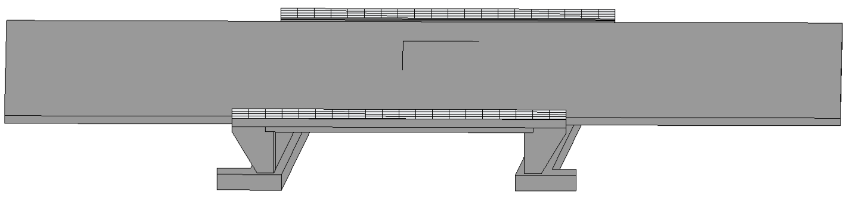
In some cases, the edges of the PPMs cannot find any point in their vicinity. Nonetheless, these edges can still be fitted into the point clouds. Note that a simple fitness function definition such as Equation 6.1 cannot provide meaningful results in these cases as the optimization algorithm cannot realize the correct placement of the passive edges.

7.4.3.3 Editability of the Resulting Model

One of the advantages of the proposed approach is the editability of the resulting model, which is required to enable design work in the frame of rehabilitation or modification measures. This feature enables users to modify each element by adjusting the value of parameters as shown in Figure 7.14. Note that a point cloud only represents the object's outer shell. For instance, it cannot provide any information about the foundation of abutments or the inner thickness of the bridge deck. Therefore, external resources are still required from which the related parameters can be extracted and imported into the model.



(a)



(b)

Figure 7.14: Editability of the model: (a) the resulting model; (b) edited model with a new span length and foundation depth.

The resulting model of the defined method preserves all the existing relationships and dependencies between elements, thanks to its parametric design. Also, all the existing parameters can be adjusted or unchanged during optimization. For example, a default value for the depth of the foundation can be assumed and remained unchanged throughout the optimization process. After optimization, this parameter can be read or extracted from structural drawings and imported into the model separately. The resulting model can also be connected to various algorithms for further enrichment. This is highly compatible with the definition of geometric DTs, which need to stay connected to the actual asset for handling bidirectional updates.

8 Conclusions

This chapter introduces significant findings in achieving the geometric representation of bridges using DTs and discusses the potential for automated processes to meet industry demands. It highlights key contributions and the current research state, addressing the limitations tackled in this thesis and those that might arise in future investigations. Furthermore, it acknowledges encountered limitations and offers insights for further refinement and investigation, as well as potential directions for future research and innovation within this domain.

This chapter encompasses three sections to conclude the thesis. The first section summarizes the proposed method introduced in the previous chapters. It reviews the concepts behind each chapter and elaborates on the thesis structure. The second section discusses the research hypotheses based on which this research has been established. It further shows how the research content can address the research hypotheses. This section further highlights the key contributions and innovations that make the research unique at its publication date. The next section is assigned to the limitations of the proposed method with a focus on each module, leading to various research topics for further investigation.

8.1 Summary of Chapters

In *Chapter 1*, entitled “Introduction”, the thesis title was divided into distinct components, including “Digital Twin (DT)”, “Bridges”, “Point Cloud Data (PCD)”, “Artificial Intelligence (AI)”, and “Optimization Algorithms”. Each term was introduced briefly, and the existing challenges, hypotheses, and requirements for creating a bridge DT were elaborated. To address these challenges, a method containing two major blocks and a middle minor module was proposed. The first module was designed to automate the bridge’s semantic segmentation process. This module receives the bridge PCD and segments it into the point cloud of bridge elements using a DL model. The middle module clusters and de-noises the segmented point clouds. It also detects boundary points and generates the final bridge dummy model required for parametric modeling. The second major module receives the clustered point clouds and the bridge dummy model. It then generates the required PPMs following the bridge 3D model (reverse engineering & parametric modeling) and solves local and global optimization problems for deriving the value of parameters from the clustered point clouds.

Chapter 2 provided a comprehensive explanation of the DT concept and its potential applications within the Architecture, Engineering, Construction, Operations, and Management (AECOM) industry, particularly in streamlining the operational and maintenance (O&M) process of bridges. This chapter illustrated how DTs can serve as a practical mechanism to support the maintenance process of existing bridges. It was discussed that a DT is defined purposefully based on a set of requirements and specifications. Requirements are the goals of the DT described based on its expected use cases, while specifications determine further details about the requirements. It was shown that at the core of an efficient bridge DT, a 3D geometric model exists whose creation, however, requires spending plenty of time and cost. PCD was introduced as one of the data sources from which the geometric bridge DT can be created. The acquisition methods for collecting PCD were elaborated and compared. Furthermore, AI techniques and optimization algorithms were explained, various modules were reviewed, and their applications in data interpretation and their potential in processing various data formats were pointed out. To shed light on the expensive steps in creating bridge DTs, the conventional approaches in generating the geometric model of bridges were explained. Additionally, the chapter highlighted the role of Internet of Things (IoT) devices in data collection and their connectivity to the DT for real-time structural assessment. It was concluded that semantic segmentation and model reconstruction are two expensive steps in manually creating bridge DTs. These particular steps were selected as the primary components within the digital twinning process, essential for automation to make the creation of DT cost-efficient.

Chapter 3 focused on the semantic segmentation and model reconstruction of bridges from point clouds. Various approaches that can be employed to automate the semantic segmentation of bridge point clouds were reviewed. It was illustrated that all existing approaches have their own advantages and drawbacks. The category bottom-up contains fundamental and practical methods to automate the semantic segmentation of bridge point clouds. However, most bottom-up methods suffer from occlusion and clutter and are dependent on setting problem-specific thresholds. Furthermore, they have been mostly defined for primitive shapes that can be expressed in a closed-form formula. Top-down methods address the occlusion problem to some extent. However, they are yet highly problem-specific and are defined considering a set of presumptions about the bridge. Therefore, they might fail in cases where presumptions are not satisfied even slightly. Deep learning (DL)-based semantic segmentation is a more recent approach that benefits from automated extraction of point features for classification. Compared to the two previous approaches, DL-based semantic segmentation provides more flexibility and adaptability with less/no dependency on problem-specific thresholds. They can also segment large-scale point clouds efficiently and pave the way for real-time evaluation.

In the second part of this chapter, the potential model reconstruction approaches to create a geometric DT were reviewed. It was explained that surface representation methods, such as triangulation and meshes, might not be appropriate to create a geometric DT due to a lack of access to geometric details such as dimensions or volumes. Hence, solid modeling approaches were elaborated. It was argued that a geometric DT must stay connected to the actual bridge through a bidirectional link to handle geometric updates; otherwise, the geometric DT concept might be compromised. It was concluded that Parametric Modeling is a solid modeling approach that provides the access point for handling geometric updates, thus providing the most appropriate representation for geometric DTs. Solid modeling methods generally need inputs to generate geometry. For instance, the implicit representation of a sphere needs input parameter values concerning the center and radius of the sphere. Therefore, the most recent approaches that provided input for solid modeling were reviewed. It was shown that most existing methods are limited to only extracting parameter values from primitive shapes that can be defined simply in a closed-form formula. It was also discussed that most bridge elements have more complicated shapes that cannot be described simply in a derivable function. At the end of this chapter, the existing research gaps that established the foundation for this research were mentioned.

Chapter 4 was assigned to elaborate on the proposed methodology for the first major module (semantic segmentation). A Multiscale Spatial Feature Descriptor network (MSFD-Net) was proposed as a DL model to automate the semantic segmentation process of bridge point clouds. It was designed to capture three sets of features, including global, local, and spatial features, and employ them in predicting the class label of points. The global features represent the individual characteristics of points in the 3D space. Local features, however, illustrate the underlying surface the points can represent in their local neighborhoods. Relative/spatial features also describe the pair-wise relationship of points in the 3D space, where global and local features cannot properly express them. MSFD-Net benefits from modules that can encode the global, local, and relative features of points. The global features are encoded using shared MLPs. Local features are described using relative position and normal deviation of points in local neighborhoods. The relative features are extracted using transformers, weighting the pairwise dependencies between points. MSFD-Net also decodes the encoded feature maps in various scales to benefit from not only high-level (fine) features generated in the last layers of the encoder but also the low-level (rough) features produced in the initial features. Besides, it summarizes the scales using an attention mechanism emphasizing the important scales in classifying points.

Chapter 5 elaborated on the middle minor module. It presented the methods required for processing segmented point clouds resulting from the first module. A clustering algorithm was proposed to automate the instance segmentation process of bridges. This algorithm is a Region Growing (RG) algorithm inspired by DBSCAN that expands regions based on the relative distance of points. Unlike DBSCAN, it benefits from a data structure and is not dependent on a threshold value for detecting noise points. To detect boundary points, the mean shift of points in each local neighborhood was calculated. Considering the higher shift of exterior points, this feature was used in the Fuzzy C-Means (FCM) clustering algorithm for classifying boundary points. To benefit from the reverse engineering approach, the 3D bridge dummy model was also generated to be used for deriving the value of parameters.

In *Chapter 6*, Parametric Prototype Models (PPMs) were proposed as tools to derive the values of parameters. PPMs are dummy parametric models defined based on a set of parameters and geometric constraints. PPMs can not only be described for primitive shapes but also more complicated geometric shapes that commonly exist in bridges. Following a reverse engineering approach and the desired final bridge model, PPMs can be designed to derive the parameter values required for creating the bridge model. To fit PPMs into the point cloud of bridge elements, a new cost function was defined that minimizes the distance of edges and vertices to points based on their contribution (activity) to the problem. This novel cost function enables the algorithm to fit PPMs into point clouds even with a large amount of occlusion. Solving the first local optimization problems leads to a set of fitted PPMs into their corresponding point clouds. To generate the integrated model of the entire bridge, PPMs need to be assembled. The assembly process of PPMs should not only generate the desired model but also retain all the parametric dependencies among the elements. For this purpose, a global optimization problem was defined to refine the fitted PPMs and integrate the shared parameters among elements. This assembly process leads to a list of parameters representing the entire bridge. Various methods for selecting PPMs were also proposed, and their use cases, advantages, and drawbacks were illustrated. Different metaheuristic algorithms were also tested for fitting PPMs into point clouds, and their performance in terms of accuracy and convergence time was compared.

Chapter 7 evaluated the performance of the proposed method in geometric digital twinning of single-span RC bridges and the bridge components mostly seen in multi-span bridges. This chapter also elaborated on the minor module and the algorithms required to process segmented point clouds and prepare them for the second module. The performance of MSFD-Net was evaluated through leave-one-out cross-validation (LOOCV) on ten bridges. In each case, various performance metrics were presented for the model. In parallel, its performance was compared with RandLA-Net (another DL model for processing point clouds).

The chapter continued with the performance evaluation of the second major module in parametric modeling of bridge components using PPMs. Two experiments were also designed to demonstrate the capabilities of the proposed method. The first experiment focused on the geometric digital twinning of ten single-span RC bridges. This experiment showed that all these bridges can be created using only four distinct PPMs, each representing a bridge element. The second experiment also tested the performance of this module in modeling multi-span bridges that are often more complicated in shape than single-span bridges. Both experiments showed the capabilities of the proposed module in model-to-cloud fitting and generating the geometric DT of the entire bridge.

This chapter further discussed the advantages of the proposed method over the conventional methods and showed its performance in scenarios with large amounts of occlusion. It also showed the model's adaptability after reconstruction, where it can still accept geometric updates to alter its shape immediately.

8.2 Contributions & Limitations

This thesis was established to address research hypotheses concerning the automated geometric digital twinning of bridges from their point clouds. The motivation behind this research stemmed from the challenges posed by traditional methods of bridge inspection, management, maintenance, and operation. The conventional methods for supporting the post-construction (as-is) phase of bridges are costly and cannot efficiently cover the large number of existing bridges. Moreover, the high costs associated with the maintenance, along with the complexities involved in data interpretation and planning, highlight the need for more efficient and reliable solutions.

By leveraging advanced technologies such as digital twinning, this research aimed to streamline the bridge O&M process by providing a comprehensive and accurate representation of bridge structures. A bridge DT serves as a virtual replica of the physical bridge, constructed from point cloud data obtained through techniques such as laser scanning or photogrammetry. This digital model not only represents the geometric details of the bridge but can also be enriched with metadata and semantics collected from the construction site.

Following the objectives outlined by *Hypothesis 1 (the O&M of bridges can be more effectively supported by DTs than by conventional management approaches)*, this research explored the potential of bridge digital twinning to support asset O&M practices by offering a more efficient and effective approach compared to conventional methods. By providing a detailed and accessible virtual representation of the bridge, well-informed decisions regarding maintenance, rehabilitation, and future planning of bridges can be made, thereby enhancing overall operational efficiency and safety.

Furthermore, the suitability of point clouds captured through photogrammetry and laser scanning for creating bridge DTs was investigated in alignment with *Hypothesis 2 (point clouds captured through laser-scanning or photogrammetry are a suitable basis for creating semantically rich DTs of existing bridges)*. Laser scanning or photogrammetry enables the generation of high-fidelity point cloud data, capturing the intricate details of the bridge structure. This data forms the foundation for creating a digital twin that accurately reflects the physical reality of the bridge, facilitating comprehensive analysis and decision-making processes.

A research method was proposed to alleviate the costs associated with the geometric digital twinning of bridges. It contained two major modules focusing on semantic segmentation and parametric modeling and an axillary/minor module to connect the major modules. The first major module introduced a novel DL model, called MSFD-Net, to automate the semantic segmentation process. This model benefits from cutting-edge AI blocks, such as transformers and various attention mechanisms, to properly describe the spatial relationship between points in a scene. It also utilizes an autoencoder with random subsampling in the network's sequential layers, making it capable of processing large-scale point clouds. Furthermore, it decodes features in various scales to generate a more enriched feature map for classifying points. These features enable the model to describe local, global, and even relative features efficiently. The results of testing this model on ten bridges show that it can reach a mean Intersection over Union (mIoU) of 91.57% in classifying points. Comparison between this model and RandLA-Net illustrated a substantial improvement by 4.88% in the mIoU value. This addresses the *Hypothesis 3 (deep neural networks allow the robust segmentation of a bridge point cloud into subsets representing individual bridge components)* by increasing the value of accuracy and providing a reliable module for automating the semantic segmentation process.

The minor module introduced a set of practical algorithms to facilitate the geometric modeling process from point clouds. This module receives the segmented point cloud of bridges from the first module and detects the required points for model-to-cloud fitting. Also, it contains the dummy parametric model of the bridge, following *Hypothesis 4 (a* *significant number of existing bridges fall into similar classes and can be represented by highly parametrized bridge models).*

The second major module introduced a research solution to derive the value of parameters from segmented point clouds. This method is based on reverse engineering and parametric modeling, where the presumed model is compared with the scanned data. Considering the desired 3D model of the bridge, PPMs are generated and used to derive the value of parameters. This step is performed by solving a piece-wise metaheuristic optimization problem to minimize the distance of points to the edges of the PPM. This optimization problem is also augmented with terms to enable the model to perform properly, even in scenarios with a large amount of occlusion, supporting *Hypothesis 5 (with the help of metaheuristic optimization approaches, pre-defined parametric model components can be fit into the respective point cloud segments).*

Contrary to conventional methods, this technique paves the way to modeling not only primitive shapes but also more complicated geometries that commonly exist in bridges. One of the considerable advantages of this approach is the exact definition of the geometries depending on the desired use cases. For instance, a rectangular PPM is necessarily defined with four orthogonal edges and two parameters controlling its width and length. This aids the model in reaching a level of abstraction that is necessary in practice. The other advantage of this approach is the adaptability of the generated models, thanks to the parametric design of PPMs. The model can receive geometric updates and change its shape depending on the value of input parameters.

The precise definition of PPMs and the governing reverse engineering approach pave the way for a structured assembly process by defining another global optimization problem on PPMs. This optimization process integrates the common parameters and generates an assembled model while preserving all the dependencies required for a parametric model. The proposed approach is the first on its own that can generate a parametric model for the entire bridge. A significant contribution of the assembly process is in its novel integration of parameters. Contrary to the conventional methods that mostly assemble the neighboring elements, this approach is capable of integrating parameters belonging to elements that stand far from each other. For instance, all piers can be assigned the same height value even though they might not be neighboring elements with shared edges. The results of testing the second module on the point cloud of ten single-span bridges show that it can model the entire bridge with a mean absolute error (MAE) of 8.43 cm. Other evaluations on the multi-span bridge elements also demonstrated the promising performance of the model-fitting method in the geometric digital twinning of bridges, supporting *Hypothesis 6 (using highly parametrized overall bridge* *models ensures the geometrically and semantically coherent creation of the DT models of existing bridges).*

In addressing the research hypotheses, the contributions highlighted in this study have been effectively realized. Through the utilization of advanced technologies and methodologies, including the development of deep learning models for semantic segmentation, the establishment of parametric modeling paradigms, and the introduction of innovative assembly methods, this research has made significant strides in automating the geometric digital twinning of bridges. These contributions not only offer practical solutions for enhancing bridge O&M but also signify advancements in the field. By bridging the gap between conventional methods and innovative approaches, this study supports conventional bridge engineering practices, offering improved efficiency, cost-effectiveness, and safety in managing and maintaining bridge infrastructure.

Considering the research results, the digital twinning process of existing bridges can be automated to a large extent. The main focus of this research was on single-span RC bridges as they form a large portion of the existing bridges in Germany (more than 50 %, according to a database received from the Federal Highway Research Institute).

MSFD-Net requires the calculation of normal vectors prior to the training process. In the current implementation, this step is conducted out of the network. However, it is highly beneficial that such features can be extracted automatically without feature engineering. Furthermore, more advanced modules can be used in the architecture, and relative/spatial features can be emphasized more to increase the network's awareness of the spatial positioning of points. The random subsampling module used in MSFD-Net makes the model capable of fast/efficient processing of large-scale point clouds. However, this subsampling strategy can result in areas with high point density being underrepresented in the sample while areas with lower density are overrepresented.

The proposed parametric modeling method might show limitations in covering more complicated bridge types, even though most of the proposed methods are extendable to such bridges; this case was shown for modeling components of multi-span bridges, however, the entire bridge model was not covered.

Apart from that, the proposed method requires bridge engineering knowledge and statistical studies to set the range of parameters that might limit the algorithm in modeling highly complicated/arbitrary shapes that are too diverse in shape. Despite having advantages in finite modeling of the bridge elements through the reverse engineering approach, it limits the model's capabilities to be applied to highly different bridges to some extent. For instance, if the input point cloud differs greatly from the presumed bridge model, the generated PPMs cannot be completely fitted into the point cloud.

This case was shown in the larger value of errors for samples with a higher difference in shape/type.

8.3 Recommendations for the Future

The promising results of this research pave the way for further research on the geometric digital twinning of bridges. The results of training a DL model on bridges show that these models can efficiently automate the semantic segmentation process of bridge point clouds. Further improvements to the model architecture and an increase in the bridge samples can potentially enhance the model's accuracy in prediction.

MSFD-Net establishes the spatial dependencies only in the bottleneck of the model on each scale before decoding. However, the spatial relationships can be expressed in each encoding block, making the model capable of an improved understanding of the scene. To this end, Global Context Blocks (GCB) can be used in the sequential layers of the network.

The subsampling strategy of the network can be improved using a pre-trained model. This model can learn how to subsample point clouds so that a higher value of accuracy can be achieved. Using this pre-trained model in the architecture can potentially improve its performance in the semantic segmentation of points.

The local features can be extracted more efficiently using other point cloud descriptors. This research employed a mechanism highly close to the Point Feature Histogram (PFH) descriptor. However, there are other types of descriptors that are commonly used for point cloud registration.

The proposed reverse engineering approach for modeling bridges requires a library or catalog of different bridge types. The ultimate goal is to have such a library containing single-span, multi-span, etc., and a classifier to select the desired model from the library in the initial step. This can be achieved using a deep learning model that receives the input point cloud and calls the desired model from the library.

As mentioned, the proposed approach needs a range of parameter values to initialize the model-fitting algorithm. To address this limitation, PPMs can be employed in a GAN model to minimize the difference between the point cloud and the generated model. However, the model proposal must be restricted to the PPM in the latent space. This approach also needs a large dataset to train the model.

Instance segmentation, denoising, and boundary point detection are all problems that can be addressed by training a neural network. If the required dataset for training can be prepared, these steps can be encapsulated into a single model without needing a minor module in the middle.

The future ideal solution can be a multi-output architecture to segment the input point cloud into the point cloud instances of bridge elements, detect boundary points or crucial points for model-fitting (using a pre-trained model), project points onto 2D planes (using normal vectors), select the proper PPM (using a classifier), and fit the PPM into the detected points (through a GAN model). Each one of these steps can be a topic for future research, and if addressed, they can make the geometric digital twining one step ahead.

Bibliography

[1] B Danette Allen. Digital Twins and Living Models at NASA, November 2021. URL https://ntrs.nasa.gov/api/citations/20210023699/downloads/ASME%20Digital%20Twin%20Summit%20Keynote_final.pdf. Document ID: 20210023699.

[2] Werner Kritzinger, Matthias Karner, Georg Traar, Jan Henjes, and Wilfried Sihn. Digital twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, 51(11):1016–1022, 2018. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2018.08.474. URL https://www.sciencedirect.com/science/article/pii/S2405896318316021. 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018.

[3] Thomas Bergs, Sascha Gierlings, Thomas Auerbach, Andreas Klink, Daniel Schraknepper, and Thorsten Augspurger. The concept of digital twin and digital shadow in manufacturing. *Procedia CIRP*, 101:81–84, 2021. ISSN 2212-8271. doi: https://doi.org/10.1016/j.procir.2021.02.010. URL https://www.sciencedirect.com/science/article/pii/S2212827121006612. 9th CIRP Conference on High Performance Cutting.

[4] Asma Ladj, Zhiqiang Wang, Oussama Meski, Farouk Belkadi, Mathieu Ritou, and Catherine Da Cunha. A knowledge-based digital shadow for machining industry in a digital twin perspective. *Journal of Manufacturing Systems*, 58:168–179, 2021. ISSN 0278-6125. doi: https://doi.org/10.1016/j.jmsy.2020.07.018. URL https://www.sciencedirect.com/science/article/pii/S027861252030128X. Digital Twin towards Smart Manufacturing and Industry 4.0.

[5] Leon Eversberg. What is a digital twin?, 2023. URL https://medium.com/geekculture/what-is-a-digital-twin-46ad1f549cce. Accessed on: 5th December 2023.

[6] Jannis Stecken, Martin Ebel, Matthias Bartelt, Jens Poeppelbuss, and Bernd Kuhlenkötter. Digital shadow platform as an innovative business model. *Pro-*

cedia CIRP, 83:204–209, 2019. ISSN 2212-8271. doi: https://doi.org/10.1016/j.procir.2019.02.130. URL `https://www.sciencedirect.com/science/article/pii/S2212827119302513`. 11th CIRP Conference on Industrial Product-Service Systems.

[7] Günther Schuh, Eric Rebentisch, Michael Riesener, Thorben Ipers, Christian Tönnes, and Merle-Hendrikje Jank. Data quality program management for digital shadows of products. *Procedia CIRP*, 86:43–48, 2019. ISSN 2212-8271. doi: https://doi.org/10.1016/j.procir.2020.01.027. URL `https://www.sciencedirect.com/science/article/pii/S2212827120300366`. 7th CIRP Global Web Conference – Towards shifted production value stream patterns through inference of data, models, and technology (CIRPe 2019).

[8] Concetta Semeraro, Mario Lezoche, Hervé Panetto, and Michele Dassisti. Digital twin paradigm: A systematic literature review. *Computers in Industry*, 130:103469, 2021. ISSN 0166-3615. doi: https://doi.org/10.1016/j.compind.2021.103469. URL `https://www.sciencedirect.com/science/article/pii/S0166361521000762`.

[9] Guodong Shao and Moneer Helu. Framework for a digital twin in manufacturing: Scope and requirements. *Manufacturing Letters*, 24:105–107, 2020. ISSN 2213-8463. doi: https://doi.org/10.1016/j.mfglet.2020.04.004. URL `https://www.sciencedirect.com/science/article/pii/S2213846319301312`.

[10] Grand View Research. Digital twin market size, share & trends analysis report by solution (component, process), by deployment (cloud, on-premise), by enterprise size, by application, by end-use, by region, and segment forecasts, 2023 - 2030, 2018-2021. URL `https://www.grandviewresearch.com/industry-analysis/digital-twin-market#`.

[11] Eric VanDerHorn and Sankaran Mahadevan. Digital twin: Generalization, characterization and implementation. *Decision Support Systems*, 145:113524, 2021. ISSN 0167-9236. doi: https://doi.org/10.1016/j.dss.2021.113524. URL `https://www.sciencedirect.com/science/article/pii/S0167923621000348`.

[12] André Borrmann, Markus König, Christian Koch, and Jakob Beetz. *Building Information Modeling: Technology Foundations and Industry Practice*. Springer, 2018. ISBN 3319928619.

[13] Bernardette Soust-Verdaguer, Carmen Llatas, and Antonio García-Martínez. Critical review of bim-based lca method to buildings. *Energy and Buildings*,

136:110–120, 2017. ISSN 0378-7788. doi: https://doi.org/10.1016/j.enbuild.2016.12.009. URL `https://www.sciencedirect.com/science/article/pii/S0378778816317650`.

[14] Stathis Eleftheriadis, Dejan Mumovic, and Paul Greening. Life cycle energy efficiency in building structures: A review of current developments and future outlooks based on BIM capabilities. *Renewable and Sustainable Energy Reviews*, 67: 811–825, 2017. doi: https://doi.org/10.1016/j.rser.2016.09.028.

[15] Graham Kelly, Michael Serginson, Steve Lockley, Nashwan Dawood, and Mohamad Kassem. BIM for facility management: a review and a case study investigating the value and challenges. In *Proceedings of the 13th International Conference on Construction Applications of Virtual Reality*, volume 5, 2013.

[16] Manish K Dixit, Varusha Venkatraj, Mohammadreza Ostadalimakhmalbaf, Fatemeh Pariafsai, and Sarel Lavy. Integration of facility management and building information modeling (bim) a review of key issues and challenges. *Facilities*, 37 (7/8):455–483, 2019. doi: https://doi.org/10.1108/F-03-2018-0043.

[17] C Koch, S German Paal, A Rashidi, Z Zhu, M König, and I Brilakis. Achievements and challenges in machine vision-based inspection of large concrete structures. *Advances in Structural Engineering*, 17(3):303–318, 2014. ISSN 1369-4332. doi: http://doi.org/10.1260/1369-4332.17.3.303.

[18] Bimal Kumar, Hubo Cai, and Makarand Hastak. An assessment of benefits of using BIM on an infrastructure project. In *International Conference on Sustainable Infrastructure*, pages 88–95, 2017. doi: http://dx.doi.org/10.1061/9780784481219.008.

[19] Brendan McGuire, Rebecca Atadero, Caroline Clevenger, and Mehmet Ozbek. Bridge information modeling for inspection and evaluation. *Journal of Bridge Engineering*, 21(4):04015076, 2016. doi: https://doi.org/10.1061/(ASCE)BE.1943-5592.0000850.

[20] Seongwoon Jeong, Rui Hou, Jerome P Lynch, Hoon Sohn, and Kincho H Law. An information modeling framework for bridge monitoring. *Advances in engineering software*, 114:11–31, 2017. doi: http://dx.doi.org/10.1016/j.advengsoft.2017.05.009.

[21] MM Marzouk and M Hisham. *Bridge information modeling in sustainable bridge management*, pages 457–466. American Society of Civil Engineers (ASCE), 2012. doi: http://dx.doi.org/10.1061/41204(426)57.

[22] CS Shim, NR Yun, and HH Song. Application of 3D bridge information modeling to design and construction of bridges. *Procedia Engineering*, 14:95–99, 2011. ISSN 1877-7058. doi: http://dx.doi.org/10.1016/j.proeng.2011.07.010.

[23] Abbas Rashidi and Ebrahim Karan. Video to BrIM: Automated 3D as-built documentation of bridges. *Journal of performance of constructed facilities*, 32(3):1–11, 2018. doi: http://dx.doi.org/10.1061/(ASCE)CF.1943-5509.0001163.

[24] Vanessa Saback de Freitas Bello, Cosmin Popescu, Thomas Blanksvärd, and Björn Täljsten. Framework for bridge management systems (BMS) using digital twins. In *Proceedings of the 1st Conference of the European Association on Quality Control of Bridges and Structures: EUROSTRUCT 2021 1*, pages 687–694. Springer, 2022. doi: https://doi.org/10.1007/978-3-030-91877-4_78.

[25] MM Futai, TN Bittencourt, RR Santos, CRR Araújo, DM Ribeiro, AR Rocha, and R Ellis. Utilization of digital twins for bridge inspection, monitoring and maintenance. In *Proceedings of the 1st Conference of the European Association on Quality Control of Bridges and Structures: EUROSTRUCT 2021 1*, pages 166–173. Springer, 2022. doi: https://doi.org/10.1007/978-3-030-91877-4_20.

[26] Brandon J. Perry, Yanlin Guo, Rebecca Atadero, and John W. van de Lindt. Streamlined bridge inspection system utilizing unmanned aerial vehicles (UAVs) and machine learning. *Measurement*, 164:108048, 2020. ISSN 0263-2241. doi: https://doi.org/10.1016/j.measurement.2020.108048.

[27] Alexis Girardet and Conrad Boton. A parametric BIM approach to foster bridge project design and analysis. *Automation in Construction*, 126:103679, 2021. doi: http://dx.doi.org/10.1016/j.autcon.2021.103679.

[28] Masoud Mohammadi, Maria Rashidi, Yang Yu, and Bijan Samali. Integration of TLS-derived Bridge Information Modeling (BrIM) with a Decision Support System (DSS) for digital twinning and asset management of bridge infrastructures. *Computers in Industry*, 147:103881, 2023. ISSN 0166-3615. doi: https://doi.org/10.1016/j.compind.2023.103881. URL `https://www.sciencedirect.com/science/article/pii/S0166361523000313`.

[29] Yuandong Pan, Zhiqi Hu, and Ioannis Brilakis. Digital Twins and Their Roles in Building Deep Renovation Life Cycle. In *Disrupting Buildings: Digitalisation and the Transformation of Deep Renovation*, pages 83–96. Springer International Publishing Cham, 2023. doi: https://doi.org/10.1007/978-3-031-32309-6.

[30] Rafael Sacks, Ioannis Brilakis, Ergo Pikas, Haiyan Sally Xie, and Mark Girolami. Construction with digital twin information systems. *Data-Centric Engineering*, 1: e14, 2020. doi: https://doi.org/10.1017/dce.2020.16.

[31] Elisa Negri, Luca Fumagalli, and Marco Macchi. A review of the roles of digital twin in CPS-based production systems. *Procedia Manufacturing*, 11:939–948, 2017. ISSN 2351-9789. doi: http://dx.doi.org/10.1016/j.promfg.2017.07.198.

[32] Federal Highway Administration. Bridge Preservation Guide Maintaining a Resilient Infrastructure to Preserve Mobility. Report, Federal Highway Administration, 2018. URL `https://www.fhwa.dot.gov/bridge/preservation/guide/guide.pdf`.

[33] National Audit Office. A Short Guide to the Department for Transport. Report, National Audit Office, 2018. URL `chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.nao.org.uk/wp-content/uploads/2015/08/Transport-short-guide1.pdf`.

[34] Mary Lou Ralls. Prefabricated bridge elements and systems in japan and europe. Report, United States. Federal Highway Administration. Office of International Programs, 2005.

[35] American Society of Civil Engineers (ASCE). ASCE's 2021 Infrastructure Report Card. Report, American Society of Civil Engineers, 2021. URL `https://infrastructurereportcard.org/`.

[36] Federal Highway Administration (FHWA). National Bridge Inspection Standards (NBIS). Rule, U.S. Department of Transportation (DOT), 2022. URL `https://www.federalregister.gov/documents/2022/05/06/2022-09512/national-bridge-inspection-standards`.

[37] Bundesministerium für Verkehr Bau und Stadtentwicklung. Bauwerksprüfung nach DIN 1076 Bedeutung, Organisation, Kosten. Report, Bundesministerium für Verkehr Bau und Stadtentwicklung, 2013. URL `https://www.bmvi.de/SharedDocs/DE/Anlage/StB/`

dokumentation-bauwerkspruefung-nach-din-1076.pdf?__blob=publicationFile.

[38] P Haardt. The german approach to bridge management: from reactive to predictive management procedures. *EUROINFRA 2009" Current State and Challenges for Sustainable Development of Infrastructure*, page 12, 2009.

[39] George Hearn. *Bridge inspection practices*, volume 375. Transportation Research Board, 2007. ISBN 978-0-309-09795-6. doi: 10.13140/RG.2.1.5179.5604.

[40] AASHTO. *The manual for bridge evaluation.* American Association of State Highway and Transportation Officials, 2018. ISBN 1560514965.

[41] Highways Agency. *Inspection manual for highway structures: Vol. 1: Reference manual*, volume 1. The Stationery Office, 2007.

[42] FHWA. *Recording and Coding Guide for the Structure Inventory and Appraisal of the Nation's Bridges. FHWA-PD-96-001.* US Department of Transportation, Federal Highway Administration, Office of Engineering Bridge Division, Wahington DC, 1995.

[43] Zanyar Mirzaei, Bryan T Adey, Leo Klatter, and P Thompson. The IABMAS bridge management committee overview of existing bridge management systems. *International Association for Bridge Maintenance and Safety (IABMAS), Sapporo, Japan*, 2014.

[44] Anweisung Straßeninformationsbank. *Teilsystem Bauwerksdaten.* BMVBS, ASB-ING, 2017. URL www.sib-bauwerke.de.

[45] Richtlinie zur einheitlichen Erfassung. *Aufzeichnung und Auswertun von Ergebnissen der Bauwerksprüfung nach DIN 1076.* BMVBS, RI-EBW-PRÜF, 2017. URL www.bast.de.

[46] Szu-Pyng Kao, Feng-Liang Wang, Jhih-Sian Lin, Jichiang Tsai, Yi-De Chu, and Pen-Shan Hung. Bridge crack inspection efficiency of an unmanned aerial vehicle system with a laser ranging module. *Sensors*, 22(12), 2022. ISSN 1424-8220. doi: 10.3390/s22124469.

[47] Iryna Osadcha, Andrius Jurelionis, and Paris Fokaides. Geometric parameter updating in digital twin of built assets: A systematic literature review. *Journal of Building Engineering*, 73:106704, 2023. ISSN 2352-7102. doi: https://doi.org/10.1016/j.jobe.2023.106704.

[48] S Vilgertshofer, MS Mafipour, A Borrmann, J Martens, T Blut, R Becker, J Blankenbach, A Glöbels, J Beetz, and F Celik. TwinGen: Advanced technologies to automatically generate digital twins for operation and maintenance of existing bridges. In *Proc. of European Conference on Product and Process Modeling*, pages 213–221, 2022. ISBN 9781003354222. doi: https://doi.org/10.1201/9781003354222.

[49] Belén Riveiro and Mercedes Solla. *Non-destructive techniques for the evaluation of structures and infrastructure*, volume 11. CRC Press, 2016.

[50] Javad Baqersad, Peyman Poozesh, Christopher Niezrecki, and Peter Avitabile. Photogrammetry and optical methods in structural dynamics–A review. *Mechanical Systems and Signal Processing*, 86:17–34, 2017. doi: 10.1016/j.ymssp.2016.02.011.

[51] Masoud Mohammadi, Maria Rashidi, Vahid Mousavi, Ali Karami, Yang Yu, and Bijan Samali. Quality evaluation of digital twins generated based on UAV photogrammetry and TLS: Bridge case study. *Remote Sensing*, 13(17):3499, 2021. doi: http://dx.doi.org/10.3390/rs13173499.

[52] Onur Özyeşil, Vladislav Voroninski, Ronen Basri, and Amit Singer. A survey of structure from motion. *Acta Numerica*, 26:305–364, 2017. doi: https://doi.org/10.1017/S096249291700006X.

[53] Dieter Fritsch and Michael Klein. 3D preservation of buildings–Reconstructing the past. *Multimedia Tools and Applications*, 77:9153–9170, 2018. doi: https://doi.org/10.1007/s11042-017-4654-5.

[54] James S Aber, Irene Marzolff, Johannes Ries, and Susan Elizabeth Ward Aber. *Small-format aerial photography and UAS imagery: Principles, techniques and geoscience applications*. Academic Press, 2019. doi: https://doi.org/10.1016/C2009-0-18493-3.

[55] Cosmin Popescu, Björn Täljsten, Thomas Blanksvärd, and Lennart Elfgren. 3D reconstruction of existing concrete bridges using optical methods. *Structure and Infrastructure Engineering*, 15(7):912–924, 2019. doi: https://doi.org/10.1080/15732479.2019.1594315.

[56] Emmanuel P. Baltsavias. A comparison between photogrammetry and laser scanning. *ISPRS Journal of Photogrammetry and Remote Sensing*, 54(2):83–94,

1999. ISSN 0924-2716. doi: https://doi.org/10.1016/S0924-2716(99)00014-3. URL https://www.sciencedirect.com/science/article/pii/S0924271699000143.

[57] Josef Jansa, Nikolaus Studnicka, Gerald Forkert, Alexander Haring, and H. Kager. Terrestrial laserscanning and photogrammetry - acquisition techniques complementing one another. 2004. URL https://api.semanticscholar.org/CorpusID:14927388.

[58] Asem Zabin, Vicente A. González, Yang Zou, and Robert Amor. Applications of machine learning to bim: A systematic literature review. *Advanced Engineering Informatics*, 51:101474, 2022. ISSN 1474-0346. doi: https://doi.org/10.1016/j.aei.2021.101474. URL https://www.sciencedirect.com/science/article/pii/S147403462100224X.

[59] Massimo Bertolini, Davide Mezzogori, Mattia Neroni, and Francesco Zammori. Machine learning for industrial applications: A comprehensive literature review. *Expert Systems with Applications*, 175:114820, 2021. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2021.114820. URL https://www.sciencedirect.com/science/article/pii/S095741742100261X.

[60] Austin G. Walters. Classify sentences via a multilayer perceptron (mlp), 2019. URL https://austingwalters.com/classify-sentences-via-a-multilayer-perceptron-mlp/. Accessed: March 14, 2024.

[61] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. doi: https://doi.org/10.48550/arXiv.1810.04805. URL http://arxiv.org/abs/1810.04805.

[62] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. doi: https://doi.org/10.48550/arXiv.1406.1078.

[63] Alex Sherstinsky. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *CoRR*, abs/1808.03314, 2018. URL http://arxiv.org/abs/1808.03314.

[64] Yu-Cheng Zhou, Zhe Zheng, Jia-Rui Lin, and Xin-Zheng Lu. Integrating nlp and context-free grammar for complex rule interpretation towards automated

compliance checking. *Computers in Industry*, 142:103746, 2022. ISSN 0166-3615. doi: https://doi.org/10.1016/j.compind.2022.103746. URL `https://www.sciencedirect.com/science/article/pii/S0166361522001439`.

[65] Chengke Wu, Xiao Li, Yuanjun Guo, Jun Wang, Zengle Ren, Meng Wang, and Zhile Yang. Natural language processing for smart construction: Current status and future directions. *Automation in Construction*, 134:104059, 2022. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2021.104059. URL `https://www.sciencedirect.com/science/article/pii/S0926580521005100`.

[66] Kasimir Forth, Jimmy Abualdenien, and André Borrmann. Calculation of embodied ghg emissions in early building design stages using bim and nlp-based semantic model healing. *Energy and Buildings*, 284:112837, 2023. ISSN 0378-7788. doi: https://doi.org/10.1016/j.enbuild.2023.112837. URL `https://www.sciencedirect.com/science/article/pii/S0378778823000671`.

[67] Songfei Wu, Qiyu Shen, Yichuan Deng, and Jack Cheng. Natural-language-based intelligent retrieval engine for bim object database. *Computers in Industry*, 108:73–88, 2019. ISSN 0166-3615. doi: https://doi.org/10.1016/j.compind.2019.02.016. URL `https://www.sciencedirect.com/science/article/pii/S0166361518304020`.

[68] Jiansong Zhang and Nora M. El-Gohary. Integrating semantic nlp and logic reasoning into a unified system for fully-automated code checking. *Automation in Construction*, 73:45–57, 2017. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2016.08.027. URL `https://www.sciencedirect.com/science/article/pii/S0926580516301819`.

[69] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. doi: https://doi.org/10.48550/arXiv.1409.1556.

[70] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. doi: https://doi.org/10.48550/arXiv.1512.03385.

[71] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. doi: https://doi.org/10.48550/arXiv.1409.4842. URL `http://arxiv.org/abs/1409.4842`.

[72] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013. doi: https://doi.org/10.48550/arXiv.1311.2524. URL http://arxiv.org/abs/1311.2524.

[73] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015. doi: https://doi.org/10.48550/arXiv.1506.02640. URL http://arxiv.org/abs/1506.02640.

[74] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. doi: https://doi.org/10.48550/arXiv.1706.03762.

[75] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018. doi: http://dx.doi.org/10.1109/CVPR.2018.00813.

[76] Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. Gcnet: Non-local networks meet squeeze-excitation networks and beyond. *CoRR*, abs/1904.11492, 2019. doi: https://doi.org/10.48550/arXiv.1904.11492. URL http://arxiv.org/abs/1904.11492.

[77] Andrew Tch. The (mostly) complete chart of neural networks explained, 2017. URL https://towardsdatascience.com/the-mostly-complete-chart-of-neural-networks-explained-3fb6f2367464. Accessed: March 14, 2024.

[78] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. doi: https://doi.org/10.48550/arXiv.1406.2661.

[79] Ming Tao, Hao Tang, Fei Wu, Xiao-Yuan Jing, Bing-Kun Bao, and Changsheng Xu. Df-gan: A simple and effective baseline for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16515–16525, 2022.

[80] Duong Huong Nguyen, Quoc Bao Nguyen, T. Bui-Tien, Guido De Roeck, and Magd Abdel Wahab. Damage detection in girder bridges using modal curvatures gapped smoothing method and convolutional neural network: Application to bo nghi bridge. *Theoretical and Applied Fracture Mechanics*, 109:102728, 2020. ISSN 0167-8442. doi: https://doi.org/10.1016/j.tafmec.2020.102728. URL `https://www.sciencedirect.com/science/article/pii/S0167844220303049`.

[81] Hadi Mahami, Navid Ghassemi, Mohammad Tayarani Darbandy, Afshin Shoeibi, Sadiq Hussain, Farnad Nasirzadeh, Roohallah Alizadehsani, Darius Nahavandi, Abbas Khosravi, and Saeid Nahavandi. Material recognition for automated progress monitoring using deep learning methods. *arXiv preprint arXiv:2006.16344*, 2020. doi: https://doi.org/10.48550/arXiv.2006.16344.

[82] Wesam Salah Alaloul and Abdul Hannan Qureshi. Material classification via machine learning techniques: construction projects progress monitoring. In *Deep Learning Applications*. IntechOpen, 2021.

[83] Nipun D. Nath, Amir H. Behzadan, and Stephanie G. Paal. Deep learning for site safety: Real-time detection of personal protective equipment. *Automation in Construction*, 112:103085, 2020. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2020.103085. URL `https://www.sciencedirect.com/science/article/pii/S0926580519308325`.

[84] Osama Mazhar, Benjamin Navarro, Sofiane Ramdani, Robin Passama, and Andrea Cherubini. A real-time human-robot interaction framework with robust background invariant hand gesture detection. *Robotics and Computer-Integrated Manufacturing*, 60:34–48, 2019. ISSN 0736-5845. doi: https://doi.org/10.1016/j.rcim.2019.05.008. URL `https://www.sciencedirect.com/science/article/pii/S0736584518302953`.

[85] Xiaolei Lv, Shengchu Zhao, Xinyang Yu, and Binqiang Zhao. Residential floor plan recognition and reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16717–16726, June 2021.

[86] Lucile Gimenez, Sylvain Robert, Frédéric Suard, and Khaldoun Zreik. Automatic reconstruction of 3d building models from scanned 2d floor plans. *Automation in Construction*, 63:48–56, 2016. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2015.12.008. URL `https://www.sciencedirect.com/science/article/pii/S0926580515002484`.

[87] Stefan F Beck, Jimmy Abualdenien, Ihab H Hijazi, André Borrmann, and Thomas H Kolbe. Analyzing contextual linking of heterogeneous information models from the domains bim and uim. *ISPRS International Journal of Geo-Information*, 10(12):807, 2021.

[88] J.J. McArthur, Nima Shahbazi, Ricky Fok, Christopher Raghubar, Brandon Bortoluzzi, and Aijun An. Machine learning and bim visualization for maintenance issue classification and enhanced data collection. *Advanced Engineering Informatics*, 38:101–112, 2018. ISSN 1474-0346. doi: https://doi.org/10.1016/j.aei.2018.06.007. URL `https://www.sciencedirect.com/science/article/pii/S1474034617305049`.

[89] Ziniu Luo and Weixin Huang. Floorplangan: Vector residential floorplan adversarial generation. *Automation in Construction*, 142:104470, 2022. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2022.104470. URL `https://www.sciencedirect.com/science/article/pii/S0926580522003430`.

[90] Francis Baek, Inhae Ha, and Hyoungkwan Kim. Augmented reality system for facility management using image-based indoor localization. *Automation in Construction*, 99:18–26, 2019. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2018.11.034. URL `https://www.sciencedirect.com/science/article/pii/S0926580518308021`.

[91] Xin-She Yang. Nature-inspired optimization algorithms: Challenges and open problems. *Journal of Computational Science*, 46:101104, 2020. ISSN 1877-7503. doi: https://doi.org/10.1016/j.jocs.2020.101104. URL `https://www.sciencedirect.com/science/article/pii/S1877750320300144`. 20 years of computational science.

[92] Yongjian Feng. Animated-gradient-descent, 2024. URL `https://github.com/MATLAB-Graphics-and-App-Building/Animated-Gradient-Descent/releases/tag/v1.1`. Retrieved March 14, 2024.

[93] Padam Singh and Sushil Kumar Choudhary. Introduction: optimization and metaheuristics algorithms. In *Metaheuristic and evolutionary computation: algorithms and applications*, pages 3–33. Springer, 2021. doi: http://dx.doi.org/10.1007/978-981-15-7571-6_1.

[94] Mahdi Shariati, Mohammad Saeed Mafipour, Behzad Ghahremani, Fazel Azarhomayun, Masoud Ahmadi, Nguyen Thoi Trung, and Ali Shariati. A novel

hybrid extreme learning machine–grey wolf optimizer (ELM-GWO) model to predict compressive strength of concrete with partial replacements for cement. *Engineering with Computers*, 32:757–779, 2020. doi: https://doi.org/10.1007/s00366-020-01081-0.

[95] Mahdi Shariati, Mohammad Saeed Mafipour, Peyman Mehrabi, Ali Shariati, Ali Toghroli, Nguyen Thoi Trung, and Musab NA Salih. A novel approach to predict shear strength of tilted angle connectors using artificial intelligence techniques. *Engineering with Computers*, 37:2089–2109, 2020. doi: https://doi.org/10.1007/s00366-019-00930-x.

[96] James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks*, volume 4, pages 1942–1948. IEEE, 1995. doi: 10.1109/ICNN.1995.488968.

[97] John H Holland. Genetic algorithms. *Scientific american*, 267(1):66–73, 1992. doi: http://dx.doi.org/10.1038/scientificamerican0792-66.

[98] R Venkata Rao, Vimal J Savsani, and DP Vakharia. Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. *Computer-aided design*, 43(3):303–315, 2011. doi: http://dx.doi.org/10.1016/j.cad.2010.12.015.

[99] Seyedali Mirjalili, Seyed Mohammad Mirjalili, and Andrew Lewis. Grey wolf optimizer. *Advances in engineering software*, 69:46–61, 2014. doi: http://dx.doi.org/10.1016/j.advengsoft.2013.12.007.

[100] Xin-She Yang. Firefly algorithms for multimodal optimization. In *International symposium on stochastic algorithms*, pages 169–178. Springer, 2009. doi: http://dx.doi.org/10.1007/978-3-642-04944-6_14.

[101] Ephramac. Visualization of particle swarm optimization (pso), 2017. URL `https://commons.wikimedia.org/wiki/File:ParticleSwarmArrowsAnimation.gif`. Licensed under CC BY-SA 4.0. Retrieved March 14, 2024.

[102] Song Ronghui and Ni Liangrong. An intelligent fuzzy-based hybrid metaheuristic algorithm for analysis the strength, energy and cost optimization of building material in construction management. *Engineering with Computers*, 38(Suppl 4): 2663–2680, 2022. doi: https://doi.org/10.1007/s00366-021-01420-9.

[103] Ryan Alberdi, Patrick Murren, and Kapil Khandelwal. Connection topology optimization of steel moment frames using metaheuristic algorithms. *Engineering Structures*, 100:276–292, 2015. ISSN 0141-0296. doi: https://doi.org/10.1016/j.engstruct.2015.06.014. URL `https://www.sciencedirect.com/science/article/pii/S0141029615003909`.

[104] Vimal J. Savsani, Ghanshyam G. Tejani, Vivek K. Patel, and Poonam Savsani. Modified meta-heuristics using random mutation for truss topology optimization with static and dynamic constraints. *Journal of Computational Design and Engineering*, 4(2):106–130, 2017. ISSN 2288-4300. doi: https://doi.org/10.1016/j.jcde.2016.10.002. URL `https://www.sciencedirect.com/science/article/pii/S2288430016300653`.

[105] S Jones, D Laquidara-Carr, A Lorenz, B Buckley, and S Barnett. The business value of BIM for infrastructure. *SmartMarket Report*, 2017.

[106] Xuanxia Yao, Jia Guo, Juan Hu, and Qixuan Cao. Using deep learning in semantic classification for point cloud data. *IEEE Access*, 7:37121–37130, 2019. doi: https://doi.org/10.1109/ACCESS.2019.2905546.

[107] Khaled El Madawi, Hazem Rashed, Ahmad El Sallab, Omar Nasr, Hanan Kamel, and Senthil Yogamani. Rgb and lidar fusion based 3d semantic segmentation for autonomous driving. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 7–12. IEEE, 2019. doi: https://doi.org/10.1109/ITSC.2019.8917447.

[108] Anestis Zaganidis, Li Sun, Tom Duckett, and Grzegorz Cielniak. Integrating deep semantic segmentation into 3-d point cloud registration. *IEEE Robotics and Automation Letters*, 3(4):2942–2949, 2018.

[109] Qingyong Hu, Bo Yang, Sheikh Khalid, Wen Xiao, Niki Trigoni, and Andrew Markham. Towards Semantic Segmentation of Urban-Scale 3D Point Clouds: A Dataset, Benchmarks and Challenges. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4977–4987, June 2021.

[110] Eran Borenstein and Shimon Ullman. Combined top-down/bottom-up segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 30(12): 2109–2125, 2008. doi: http://dx.doi.org/10.1109/TPAMI.2007.70840.

[111] Jorge J Palop, Lennart Mucke, and Erik D Roberson. Quantifying biomarkers of cognitive dysfunction and neuronal network hyperexcitability in mouse models of Alzheimer's disease: depletion of calcium-dependent proteins and inhibitory hippocampal remodeling. In *Alzheimer's Disease and Frontotemporal Dementia*, pages 245–262. Springer, 2010. doi: http://dx.doi.org/10.1007/978-1-60761-744-0_17.

[112] Aparajithan Sampath and Jie Shan. Segmentation and reconstruction of polyhedral building roofs from aerial lidar point clouds. *IEEE Transactions on geoscience and remote sensing*, 48(3):1554–1567, 2009. doi: http://dx.doi.org/10.1109/TGRS.2009.2030180.

[113] Zoltan Csaba Marton, Radu Bogdan Rusu, and Michael Beetz. On fast surface reconstruction methods for large and noisy point clouds. In *2009 IEEE international conference on robotics and automation*, pages 3218–3223. IEEE, 2009. doi: http://dx.doi.org/10.1109/ROBOT.2009.5152628.

[114] Cheng Zhang and Pingbo Tang. Visual complexity analysis of sparse imageries for automatic laser scan planning in dynamic environments. *Computing in Civil Engineering*, pages 271–279, 2015. doi: http://dx.doi.org/10.1061/9780784479247.034.

[115] Andrey Dimitrov, Rongqi Gu, and Mani Golparvar-Fard. 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, 2016. doi: http://dx.doi.org/10.1111/mice.12192.

[116] Andrey Dimitrov and Mani Golparvar-Fard. Segmentation of building point cloud models including detailed architectural/structural features and MEP systems. *Automation in Construction*, 51:32–45, 2015. doi: http://dx.doi.org/10.1016/j.autcon.2014.12.015.

[117] Paul J. Besl and Ramesh C Jain. Segmentation through variable-order surface fitting. *IEEE Transactions on pattern analysis and machine intelligence*, 10(2):167–192, 1988. doi: http://doi.org/10.1109/34.3881.

[118] V.F. Leavers. Which Hough Transform? *CVGIP: Image Understanding*, 58(2):250–264, 1993. ISSN 1049-9660. doi: https://doi.org/10.1006/ciun.1993.1041. URL `https://www.sciencedirect.com/science/article/pii/S1049966083710417`.

[119] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[120] Dirk Hähnel, Wolfram Burgard, and Sebastian Thrun. Learning compact 3d models of indoor and outdoor environments with a mobile robot. *Robotics and Autonomous Systems*, 44(1):15–27, 2003. doi: https://doi.org/10.1016/S0921-8890(03)00007-1.

[121] Shweta Kansal and Pradeep Jain. Automatic seed selection algorithm for image segmentation using region growing. *International Journal of Advances in Engineering & Technology*, 8(3):362, 2015.

[122] Kazuaki Kawashima, Satoshi Kanai, and Hiroaki Date. 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, 2014. doi: http://doi.org/10.7315/JCDE.2014.002.

[123] Hélène Macher, Tania Landes, and Pierre Grussenmeyer. From point clouds to building information models: 3D semi-automatic reconstruction of indoors of existing buildings. *Applied Sciences*, 7(10):1030, 2017. doi: http://doi.org/10.3390/app7101030.

[124] Ruwen Schnabel, Roland Wahl, and Reinhard Klein. Efficient RANSAC for point-cloud shape detection. In *Computer graphics forum*, volume 26, pages 214–226. Wiley Online Library, 2007. doi: http://dx.doi.org/10.1111/j.1467-8659.2007.01016.x.

[125] Ramy Ashraf Zeineldin and Nawal Ahmed El-Fishawy. A Survey of RANSAC enhancements for Plane Detection in 3D Point Clouds. *Menoufia Journal of Electronic Engineering Research*, 26(2):519–537, 2017. doi: https://doi.org/10.21608/mjeer.2017.63627.

[126] J. Illingworth and J. Kittler. A survey of the hough transform. *Computer Vision, Graphics, and Image Processing*, 44(1):87–116, 1988. ISSN 0734-189X. doi: https://doi.org/10.1016/S0734-189X(88)80033-1. URL `https://www.sciencedirect.com/science/article/pii/S0734189X88800331`.

[127] Srivathsan Murali, Pablo Speciale, Martin R Oswald, and Marc Pollefeys. Indoor Scan2BIM: Building information models of house interiors. In *2017 IEEE/RSJ*

International Conference on Intelligent Robots and Systems (IROS), pages 6126–6133. IEEE, 2017. doi: https://doi.org/10.1109/IROS.2017.8206513.

[128] Hakim Boulaassal, Tania Landes, Pierre Grussenmeyer, and Fayez Tarsha-Kurdi. Automatic segmentation of building facades using terrestrial laser data. In *ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007*, pages 65–70, 2007.

[129] Rostislav Hulik, Michal Spanel, Pavel Smrz, and Zdenek Materna. Continuous plane detection in point-cloud data based on 3d hough transform. *Journal of Visual Communication and Image Representation*, 25(1):86–97, 2014. ISSN 1047-3203. doi: https://doi.org/10.1016/j.jvcir.2013.04.001. URL `https://www.sciencedirect.com/science/article/pii/S104732031300062X`. Visual Understanding and Applications with RGB-D Cameras.

[130] Jae Hyuk Lee, Jeong Jun Park, and Hyungchul Yoon. Automatic bridge design parameter extraction for scan-to-BIM. *Applied Sciences*, 10(20):7346, 2020. doi: http://dx.doi.org/10.3390/app10207346.

[131] Philip HS Torr and Andrew Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *Computer vision and image understanding*, 78(1):138–156, 2000.

[132] Yujie Yan and Jerome F. Hajjar. Geometric models from laser scanning data for superstructure components of steel girder bridges. *Automation in Construction*, 142:104484, 2022. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2022.104484.

[133] Siyuan Chen, Linh Truong-Hong, Debra Laefer, and Eleni Mangina. Automated bridge deck evaluation through UAV derived point cloud. In *CERI-ITRN2018*, pages 735–740, 2018.

[134] Linh Truong-Hong and Roderik Lindenbergh. Automatically extracting surfaces of reinforced concrete bridges from terrestrial laser scanning point clouds. *Automation in Construction*, 135:104127, 2022. doi: http://dx.doi.org/10.1016/j.autcon.2021.104127.

[135] Viorica Pătrăucean, Iro Armeni, Mohammad Nahangi, Jamie Yeung, Ioannis Brilakis, and Carl Haas. State of research in automatic as-built modelling. *Advanced Engineering Informatics*, 29(2):162–171, 2015. doi: https://doi.org/10.1016/j.aei.2015.01.001.

[136] Iasonas Kokkinos, Petros Maragos, and Alan Yuille. Bottom-up & top-down object detection using primal sketch features and graphical models. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1893–1900. IEEE, 2006. doi: http://dx.doi.org/10.1109/CVPR.2006.74.

[137] Oladipupo Esau Taiwo, Abikoye Oluwakemi Christianah, Akande Noah Oluwatobi, Kayode Anthonia Aderonke, and Adeniyi Jide kehinde. Comparative study of two divide and conquer sorting algorithms: Quicksort and mergesort. *Procedia Computer Science*, 171:2532–2540, 2020. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2020.04.274. URL `https://www.sciencedirect.com/science/article/pii/S1877050920312667`. Third International Conference on Computing and Network Communications (CoCoNet'19).

[138] Conor Dore and Maurice Murphy. Semi-automatic generation of as-built BIM façade geometry from laser and image data. *Journal of Information Technology in Construction (ITcon)*, 19(2):20–46, 2014. ISSN 1874-4753.

[139] Shi Pu and George Vosselman. Knowledge based reconstruction of building models from terrestrial laser scanning data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 64(6):575–584, 2009. doi: http://dx.doi.org/10.1016/j.isprsjprs.2009.04.001.

[140] Hema Koppula, Abhishek Anand, Thorsten Joachims, and Ashutosh Saxena. Semantic labeling of 3d point clouds for indoor scenes. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

[141] Mahmoud Fouad Ahmed, Carl T Haas, and Ralph Haas. Automatic detection of cylindrical objects in built facilities. *Journal of Computing in Civil Engineering*, 28(3):04014009, 2014. doi: http://dx.doi.org/10.1061/(ASCE)CP.1943-5487.0000329.

[142] WB Thompson and JC Owen. Feature-based reverse engineering of mechanical parts. *IEEE Transactions on robotics and automation*, 15(1):57–66, 1999. doi: http://dx.doi.org/10.1109/70.744602.

[143] Ruodan Lu, Ioannis Brilakis, and Campbell R Middleton. Detection of structural components in point clouds of existing RC bridges. *Computer-Aided Civil and*

Infrastructure Engineering, 34(3):191–212, 2019. doi: http://dx.doi.org/10.1111/mice.12407.

[144] Yi-Pu Zhao, Haotian Wu, and Patricio A Vela. *Top-down partitioning of reinforced concrete bridge components*, pages 275–283. American Society of Civil Engineers Reston, VA, 2019. doi: 10.1061/9780784482445.035.

[145] Yujie Yan and Jerome F Hajjar. Automated extraction of structural elements in steel girder bridges from laser point clouds. *Automation in Construction*, 125: 103582, 2021. doi: http://dx.doi.org/10.1016/j.autcon.2021.103582.

[146] Yue Pan, Yiqing Dong, Dalei Wang, Airong Chen, and Zhen Ye. Three-dimensional reconstruction of structural surface model of heritage bridges using UAV-based photogrammetric point clouds. *Remote Sensing*, 11(10):1204, 2019. doi: http://dx.doi.org/10.3390/rs11101204.

[147] Guocheng Qin, Yin Zhou, Kaixin Hu, Daguang Han, and Chunli Ying. Automated reconstruction of parametric bim for bridge based on terrestrial laser scanning data. *Advances in Civil Engineering*, 2021:1, 2021. doi: http://dx.doi.org/10.1155/2021/8899323.

[148] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. ISSN 07300301. doi: http://dx.doi.org/10.1145/3326362.

[149] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4490–4499, 2018. doi: http://dx.doi.org/10.1109/CVPR.2018.00472.

[150] Jin Xie, Yi Fang, Fan Zhu, and Edward Wong. Deepshape: Deep learned shape descriptor for 3d shape matching and retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1275–1283, 2015. doi: http://dx.doi.org/10.1109/TPAMI.2016.2596722.

[151] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015. doi: http://dx.doi.org/10.1109/ICCV.2015.114.

[152] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[153] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. doi: https://doi.org/10.48550/arXiv.1706.02413.

[154] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015. doi: 10.1007/978-3-319-24574-4_28.

[155] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4558–4567, 2018. doi: http://dx.doi.org/10.1109/CVPR.2018.00479.

[156] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16259–16268, 2021. doi: http://dx.doi.org/10.1109/ICCV48922.2021.01595.

[157] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6411–6420, 2019. doi: https://doi.org/10.48550/arXiv.1904.08889.

[158] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11108–11117, 2020. doi: http://dx.doi.org/10.1109/CVPR42600.2020.01112.

[159] Haoxi Ran, Jun Liu, and Chengjie Wang. Surface representation for point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern*

Recognition, pages 18942–18952, 2022. doi: https://doi.org/10.48550/arXiv.2205.05740.

[160] Lingxiao Li, Minhyuk Sung, Anastasia Dubrovina, Li Yi, and Leonidas J Guibas. Supervised fitting of geometric primitives to 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2652–2660, 2019. doi: http://dx.doi.org/10.1109/CVPR.2019.00276.

[161] Fangqiao Hu, Jin Zhao, Yong Huang, and Hui Li. Structure-aware 3D reconstruction for cable-stayed bridges: A learning-based method. *Computer-Aided Civil and Infrastructure Engineering*, 36(1):89–108, 2021. doi: http://dx.doi.org/10.1111/mice.12568.

[162] Jun S Lee, Jeongjun Park, and Young-Moo Ryu. Semantic segmentation of bridge components based on hierarchical point cloud model. *Automation in Construction*, 130:103847, 2021. doi: http://dx.doi.org/10.1016/j.autcon.2021.103847.

[163] Tian Xia, Jian Yang, and Long Chen. Automated semantic segmentation of bridge point cloud based on local descriptor and machine learning. *Automation in Construction*, 133:103992, 2022. doi: http://dx.doi.org/10.1016/j.autcon.2021.103992.

[164] Yixiong Jing, Brian Sheil, and Sinan Acikgoz. Segmentation of large-scale masonry arch bridge point clouds with a synthetic simulator and the BridgeNet neural network. *Automation in Construction*, 142:104459, 2022. doi: http://dx.doi.org/10.1016/j.autcon.2022.104459.

[165] Xiaofei Yang, Enrique del Rey Castillo, Yang Zou, Liam Wotherspoon, and Yi Tan. Automated semantic segmentation of bridge components from large-scale point clouds using a weighted superpoint graph. *Automation in Construction*, 142:104519, 2022. doi: http://dx.doi.org/10.1016/j.autcon.2022.104519.

[166] Xiaofei Yang, Enrique del Rey Castillo, Yang Zou, and Liam Wotherspoon. Semantic segmentation of bridge point clouds with a synthetic data augmentation strategy and graph-structured deep metric learning. *Automation in Construction*, 150:104838, 2023. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2023.104838. URL `https://www.sciencedirect.com/science/article/pii/S0926580523000985`.

[167] John Vince, Vince, and DESMOND. *Calculus for computer graphics*, volume 112. Springer, 2013. ISBN 978-3-031-28117-4. doi: https://doi.org/10.1007/978-3-031-28117-4.

[168] 3d meshes with leica cyclone 3dr. https://leica-geosystems.com/de-de/products/laser-scanners/software/leica-cyclone/leica-cyclone-3dr/3d-meshes-with-cyclone-3dr. Accessed: 2023-10-30.

[169] Sven Oesau, Florent Lafarge, and Pierre Alliez. Indoor scene reconstruction using feature sensitive primitive extraction and graph-cut. *ISPRS journal of photogrammetry and remote sensing*, 90:68–82, 2014. doi: http://dx.doi.org/10.1016/j.isprsjprs.2014.02.004.

[170] Vladimir Logvinovich Rvachev. An analytic description of certain geometric objects. In *Doklady Akademii Nauk*, volume 153, pages 765–767. Russian Academy of Sciences, 1963.

[171] A Ricci. A constructive geometry for computer graphics. *Computer-Aided Design*, 6(1):53–53, 1974.

[172] J. Stewart. *Multivariable Calculus: Concepts and Contexts*. Available 2010 Titles Enhanced Web Assign Series. Cengage Learning, 2009. ISBN 9780495560548. URL https://books.google.de/books?id=uYxE7bzhh08C.

[173] Xinghua Song and Bert Jüttler. Modeling and 3d object reconstruction by implicitly defined surfaces with sharp features. *Computers & Graphics*, 33(3):321–330, 2009.

[174] Kevin J Weiler. *Topological Structures For Geometric Modeling (Boundary Representation, Manifold, Radial Edge Structure)*. Rensselaer Polytechnic Institute, 1986.

[175] André Borrmann, Markus König, Christian Koch, and Jakob Beetz. Building information modeling : Why ? what ? how ?, 2018.

[176] Ashok Kumar Patil, Pavitra Holi, Sang Keun Lee, and Young Ho Chai. An adaptive approach for the reconstruction and modeling of as-built 3d pipelines from point clouds. *Automation in construction*, 75:65–78, 2017.

[177] Captain Sprite. Boolean difference operation on a cube and a sphere, 2006. URL https://en.wikipedia.org/wiki/File:Boolean_difference.PNG. Uploaded May 16, 2023. Licensed under CC BY-SA 3.0. Retrieved: March 14, 2024.

[178] Junxiang Zhu, Peng Wu, Mengcheng Chen, Mi Jeong Kim, Xiangyu Wang, and Tingchen Fang. Automatically processing IFC clipping representation for BIM

and GIS integration at the process level. *Applied sciences*, 10(6):2009, 2020. doi: https://doi.org/10.3390/app10062009.

[179] Huang Ming, Du Yanzhu, Zhang Jianguang, and Zhang Yong. A topological enabled three-dimensional model based on constructive solid geometry and boundary representation. *Cluster Computing*, 19:2027–2037, 2016. doi: https://doi.org/10.1007/s10586-016-0634-1.

[180] Umit Isikdag and Sisi Zlatanova. Towards defining a framework for automatic generation of buildings in citygml using building information models. In *3D geo-information sciences*, pages 79–96. Springer, 2009. doi: https://doi.org/10.1007/978-3-540-87395-2_6.

[181] Duhwan Mun, Soonhung Han, Junhwan Kim, and Youchon Oh. A set of standard modeling commands for the history-based parametric approach. *Computer-Aided Design*, 35(13):1171–1179, 11 2003. ISSN 0010-4485. doi: http://dx.doi.org/10.1016/S0010-4485(03)00022-8.

[182] Javier Monedero. Parametric design: a review and some experiences. *Automation in construction*, 9(4):369–377, 2000. doi: http://dx.doi.org/10.1016/S0926-5805(99)00020-5.

[183] Jami J. Shah and Martti Mäntylä. *Parametric and feature-based CAD/CAM: Concepts, Techniques and Applications*. ohn Wiley & Sons, New York, 1995. ISBN 0-471-00214-3.

[184] Yang Ji, André Borrmann, Jakob Beetz, and Mathias Obergrießer. Exchange of Parametric Bridge Models Using a Neutral Data Format. *Journal of Computing in Civil Engineering*, 27(6):593–606, 11 2013. ISSN 0887-3801. doi: 10.1061/(ASCE)CP.1943-5487.0000286.

[185] Carl Schultz, Mehul Bhatt, and André Borrmann. Bridging qualitative spatial constraints and feature-based parametric modelling: Expressing visibility and movement constraints. *Advanced Engineering Informatics*, 31:2–17, 11 2015. ISSN 14740346. doi: 10.1016/j.aei.2015.10.004.

[186] MS Mafipour, S Vilgertshofer, and A Borrmann. Digital twinning of bridges from point cloud data by deep learning and parametric models. In *Proc. of European Conference on Product and Process Modeling 2022*, 2022.

[187] Jun Wang, Dongxiao Gu, Zeyun Yu, Changbai Tan, and Laishui Zhou. A framework for 3d model reconstruction in reverse engineering. *Computers & Industrial Engineering*, 63(4):1189–1200, 2012. ISSN 0360-8352. doi: https://doi.org/10.1016/j.cie.2012.07.009. URL `https://www.sciencedirect.com/science/article/pii/S0360835212001842`.

[188] Yangyan Li, Xiaokun Wu, Yiorgos Chrysathou, Andrei Sharf, Daniel Cohen-Or, and Niloy J. Mitra. Globfit: Consistently fitting primitives by discovering global relations. In *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450309431. doi: 10.1145/1964921.1964947. URL `https://doi.org/10.1145/1964921.1964947`.

[189] Roseline Bénière, Gérard Subsol, Gilles Gesquière, François Le Breton, and William Puech. A comprehensive process of reverse engineering from 3D meshes to CAD models. *Computer-Aided Design*, 45(11):1382–1393, 2013. doi: https://doi.org/10.1016/j.cad.2013.06.004.

[190] Tamas Varady, Ralph R Martin, and Jordan Cox. Reverse engineering of geometric models—an introduction. *Computer-aided design*, 29(4):255–268, 1997. doi: https://doi.org/10.1016/S0010-4485(96)00054-1.

[191] Francesco Buonamici, Monica Carfagni, Rocco Furferi, Lapo Governi, Alessandro Lapini, and Yary Volpe. Reverse engineering modeling methods and tools: a survey. *Computer-Aided Design and Applications*, 15(3):443–464, 2018. doi: https://doi.org/10.1080/16864360.2017.1397894.

[192] Alexandre Durupt, Sebastien Remy, Guillaume Ducellier, and Benoît Eynard. From a 3D point cloud to an engineering CAD model: a knowledge-product-based approach for reverse engineering. *Virtual and Physical Prototyping*, 3(2):51–59, 2008. doi: http://dx.doi.org/10.1080/17452750802047917.

[193] Hyungki Kim, Changmo Yeo, Inhwan Dennis Lee, and Duhwan Mun. Deep-learning-based retrieval of piping component catalogs for plant 3D CAD model reconstruction. *Computers in Industry*, 123:103320, 2020. ISSN 0166-3615. doi: https://doi.org/10.1016/j.compind.2020.103320.

[194] Tahir Rabbani and Frank Van Den Heuvel. Efficient hough transform for automatic detection of cylinders in point clouds. *Proc ISPRS Workshop Laser Scan 2005, ISPRS Arch*, 36:60–65, 2005.

[195] Tahir Rabbani. Automatic reconstruction of industrial installations using point clouds and images. Report, TU Delft: Faculty of Civil Engineering and Geosciences, 2006. URL `http://resolver.tudelft.nl/uuid:0012068e-93b4-4bd9-a9b3-9c579ae7c91a`.

[196] Franco P. Preparata and Se June Hong. Convex hulls of finite sets of points in two and three dimensions. *Communications of the ACM*, 20(2):87–93, 1977. doi: http://dx.doi.org/10.1145/359423.359430.

[197] Herbert Edelsbrunner and Ernst P Mücke. Three-dimensional alpha shapes. *ACM Transactions On Graphics (TOG)*, 13(1):43–72, 1994. doi: https://doi.org/10.1145/174462.156635.

[198] Matt Duckham, Lars Kulik, Mike Worboys, and Antony Galton. Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. *Pattern Recognition*, 41(10):3224–3236, 2008. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2008.03.023.

[199] Adriano Moreira and Maribel Yasmina Santos. Concave hull: A k-nearest neighbours approach for the computation of the region occupied by a set of points. In *Proceedings of the Second International Conference on Computer Graphics Theory and Applications*, pages 61–68. INSTICC Press, 2007. ISBN 978-972-8865-71-9. doi: 10.5220/0002080800610068.

[200] Nina Amenta, Marshall Bern, and David Eppstein. The Crust and the β-Skeleton: Combinatorial Curve Reconstruction. *Graphical Models and Image Processing*, 60 (2):125–135, 1998. ISSN 1077-3169. doi: https://doi.org/10.1006/gmip.1998.0465.

[201] Ruodan Lu and Ioannis Brilakis. Digital twinning of existing reinforced concrete bridges from labelled point clusters. *Automation in Construction*, 105:102837, 2019. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2019.102837.

[202] Guangcong Zhang, Patricio A Vela, Peter Karasev, and Ioannis Brilakis. 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, 2015. doi: http://dx.doi.org/10.1111/mice.12063.

[203] Boyu Wang, Chao Yin, Han Luo, Jack CP Cheng, and Qian Wang. Fully automated generation of parametric BIM for MEP scenes based on terrestrial laser scanning data. *Automation in Construction*, 125:103615, 2021. doi: http://dx.doi.org/10.1016/j.autcon.2021.103615.

[204] Liu Yang, Jack C.P. Cheng, and Qian Wang. Semi-automated generation of parametric BIM for steel structures based on terrestrial laser scanning data. *Automation in Construction*, 112:103037, 2020. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2019.103037.

[205] Soon-Wook Kwon, Frederic Bosche, Changwan Kim, Carl T. Haas, and Katherine A. Liapi. Fitting range data to primitives for rapid local 3D modeling using sparse range point clouds. *Automation in Construction*, 13(1):67–81, 2004. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2003.08.007. The best of ISARC 2002.

[206] Andrés Justo, Daniel Lamas, Ana Sánchez-Rodríguez, Mario Soilán, and Belén Riveiro. Generating IFC-compliant models and structural graphs of truss bridges from dense point clouds. *Automation in Construction*, 149:104786, 2023. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2023.104786.

[207] Enrique Valero, Antonio Adán, and Carlos Cerrada. Automatic method for building indoor boundary models from dense point clouds collected by laser scanners. *Sensors*, 12(12):16099–16115, 2012. doi: http://dx.doi.org/10.3390/s121216099.

[208] Ashok Kumar Patil, Pavitra Holi, Sang Keun Lee, and Young Ho Chai. An adaptive approach for the reconstruction and modeling of as-built 3D pipelines from point clouds. *Automation in Construction*, 75:65–78, 2017. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2016.12.002.

[209] Sara B Walsh, Daniel J Borello, Burcu Guldur, and Jerome F Hajjar. Data processing of point clouds for object detection for structural engineering applications. *Computer-Aided Civil and Infrastructure Engineering*, 28(7):495–508, 2013. doi: http://dx.doi.org/10.1111/mice.12016.

[210] Debra F. Laefer and Linh Truong-Hong. Toward automatic generation of 3D steel structures for building information modelling. *Automation in Construction*, 74: 66–77, 2017. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2016.11.011.

[211] Luigi Barazzetti. Parametric as-built model generation of complex shapes from point clouds. *Advanced Engineering Informatics*, 30(3):298–311, 2016. ISSN 1474-0346. doi: https://doi.org/10.1016/j.aei.2016.03.005.

[212] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. Aligning point cloud views using persistent feature histograms. In *2008 IEEE/RSJ*

international conference on intelligent robots and systems, pages 3384–3391. IEEE, 2008. doi: http://dx.doi.org/10.1109/IROS.2008.4650967.

[213] Kangcheng Liu, Zhi Gao, Feng Lin, and Ben M Chen. Fg-net: Fast large-scale lidar point clouds understanding network leveraging correlated feature mining and geometric-aware modelling. *arXiv preprint arXiv:2012.09439*, 2020. doi: https://doi.org/10.48550/arXiv.2012.09439.

[214] Shi Qiu, Saeed Anwar, and Nick Barnes. Semantic segmentation for real point cloud scenes via bilateral augmentation and adaptive fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1757–1767, 2021. doi: https://arxiv.org/abs/2103.07074v2.

[215] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[216] Syeda Mariam Ahmed, Yan Zhi Tan, Chee Meng Chew, Abdullah Al Mamun, and Fook Seng Wong. Edge and corner detection for unorganized 3d point clouds with application to robotic welding. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7350–7355. IEEE, 2018. doi: http://dx.doi.org/10.1109/IROS.2018.8593910.

[217] James C Bezdek. *Pattern recognition with fuzzy objective function algorithms*. Springer Science & Business Media, 2013. ISBN 1-4757-0450-1. doi: http://dx.doi.org/10.1007/978-1-4757-0450-1.

[218] Murat Arikan, Michael Schwärzler, Simon Flöry, Michael Wimmer, and Stefan Maierhofer. O-snap: Optimization-based snapping for modeling architecture. *ACM Transactions on Graphics (TOG)*, 32(1):1–15, 2013. doi: http://dx.doi.org/10.1145/2421636.2421642.

[219] Wei Sui, Lingfeng Wang, Bin Fan, Hongfei Xiao, Huaiyu Wu, and Chunhong Pan. Layer-wise floorplan extraction for automatic urban building reconstruction. *IEEE transactions on visualization and computer graphics*, 22(3):1261–1277, 2015. doi: http://dx.doi.org/10.1109/TVCG.2015.2505296.

[220] Zong Woo Geem, Joong Hoon Kim, and Gobichettipalayam Vasudevan Loganathan. A new heuristic optimization algorithm: harmony search. *simulation*, 76(2):60–68, 2001. doi: https://doi.org/10.1177/003754970107600201.

[221] A Kai Qin, Vicky Ling Huang, and Ponnuthurai N Suganthan. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE transactions on Evolutionary Computation*, 13(2):398–417, 2008. doi: https://doi.org/10.1109/TEVC.2008.927706.

[222] Bo Xing, Wen-Jing Gao, Bo Xing, and Wen-Jing Gao. Invasive weed optimization algorithm. *Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms*, 62:177–181, 2014. doi: https://doi.org/10.1007/978-3-319-03404-1_13.

[223] Muzaffar Eusuff, Kevin Lansey, and Fayzul Pasha. Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. *Engineering optimization*, 38 (2):129–154, 2006. doi: https://doi.org/10.1080/03052150500384759.

[224] Peter JM Van Laarhoven and Emile HL Aarts. *Simulated Annealing: Theory and Applications*, volume 1. Springer Dordrecht, 1987. ISBN 978-90-277-2513-4. doi: https://doi.org/10.1007/978-94-015-7744-1.

[225] Harish Garg. A hybrid PSO-GA algorithm for constrained optimization problems. *Applied Mathematics and Computation*, 274:292–305, 2016. doi: https://doi.org/10.1016/j.amc.2015.11.001.

[226] Agisoft Metashape Professional. `https://www.agisoft.com/`. Accessed: March 14, 2024.

[227] Ruodan Lu, Ioannis Brilakis, and Campbell R Middleton. Detection of structural components in point clouds of existing RC bridges. *Computer-Aided Civil and Infrastructure Engineering*, 34(3):191–212, 2019. ISSN 1093-9687. doi: http://dx.doi.org/10.1111/mice.12407.

[228] Iro Armeni, Ozan Sener, Amir R. Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[229] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9297–9307, 2019.

[230] Weikai Tan, Nannan Qin, Lingfei Ma, Ying Li, Jing Du, Guorong Cai, Ke Yang, and Jonathan Li. Toronto-3d: A large-scale mobile lidar dataset for semantic

segmentation of urban roadways. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

[231] Daniel Lamas, Andrés Justo, Mario Soilán, Manuel Cabaleiro, and Belén Riveiro. Instance and semantic segmentation of point clouds of large metallic truss bridges. *Automation in Construction*, 151:104865, 2023. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2023.104865.

[232] B. Riveiro, M.J. DeJong, and B. Conde. Automated processing of large point clouds for structural health monitoring of masonry arch bridges. *Automation in Construction*, 72:258–268, 2016. ISSN 0926-5805. doi: https://doi.org/10.1016/j.autcon.2016.02.009.

[233] Saeed Asaeedi, Farzad Didehvar, and Ali Mohades. α-Concave hull, a generalization of convex hull. *Theoretical Computer Science*, 702:48–59, 2017. ISSN 0304-3975. doi: https://doi.org/10.1016/j.tcs.2017.08.014.

A Appendix I - Semantic Segmentation

Table A.1 compares the prediction results of the models in terms of Precision. MSFD-Net demonstrates a precision range spanning from 89% to 99%. This range signifies the model's consistent and high-accuracy identification capability. Notably, it achieves

Table A.1: Comparative results of Precision throughout the LOOCV test.

Sample	Model	Background	Deck	Abutment	Railing	Avg.
Bridge 01	MSFD-Net	99.11	95.49	96.69	85.76	94.26
	RandLA-Net	99.34	90.15	95.12	81.52	91.53
Bridge 02	MSFD-Net	99.31	95.90	94.43	79.96	92.40
	RandLA-Net	98.58	95.92	94.98	62.98	88.12
Bridge 03	MSFD-Net	98.36	97.21	96.29	83.91	93.94
	RandLA-Net	95.74	94.52	93.08	64.79	87.03
Bridge 04	MSFD-Net	99.81	96.58	96.03	89.75	95.54
	RandLA-Net	99.47	94.97	93.55	80.44	92.11
Bridge 05	MSFD-Net	95.21	99.39	93.82	88.83	94.31
	RandLA-Net	98.69	94.70	92.46	82.69	92.14
Bridge 06	MSFD-Net	98.45	97.83	97.03	81.09	93.60
	RandLA-Net	99.58	96.00	94.66	70.64	90.22
Bridge 07	MSFD-Net	99.72	97.46	96.50	95.23	97.23
	RandLA-Net	99.23	98.77	91.98	80.98	92.74
Bridge 08	MSFD-Net	95.02	99.67	97.67	90.42	95.70
	RandLA-Net	94.03	97.36	94.52	72.69	89.65
Bridge 09	MSFD-Net	98.93	93.66	93.48	92.58	94.66
	RandLA-Net	99.00	94.79	87.27	90.91	92.99
Bridge 10	MSFD-Net	89.96	98.85	96.03	92.20	94.26
	RandLA-Net	95.59	99.09	91.50	75.63	90.45

higher precision in Deck and Background components, suggesting a superior ability to segment these structural elements. This may be attributed to simpler contextual variations present in these components compared to Abutment and Railing. In contrast, RandLA-Net displays a wider precision range, spanning from 75% to 99%. This broader variability indicates fluctuations in the model's precision across components. The model tends to exhibit lower precision compared to MSFD-Net, particularly noticeable in Abutment and Railing components.

Table A.2: Comparative results of Recall throughout the LOOCV test.

Sample	Model	Background	Deck	Abutment	Railing	Avg.
Bridge 01	MSFD-Net	96.85	98.33	99.44	98.77	98.35
	RandLA-Net	92.92	98.49	99.87	96.54	96.96
Bridge 02	MSFD-Net	95.70	99.07	99.48	93.15	96.85
	RandLA-Net	95.60	97.82	99.27	99.17	97.97
Bridge 03	MSFD-Net	97.86	97.05	99.68	96.34	97.73
	RandLA-Net	95.57	92.91	99.88	93.14	95.37
Bridge 04	MSFD-Net	96.51	99.40	99.43	98.82	98.54
	RandLA-Net	94.83	98.26	99.82	99.21	98.03
Bridge 05	MSFD-Net	99.25	93.63	99.31	85.63	94.45
	RandLA-Net	94.82	97.65	99.75	94.15	96.59
Bridge 06	MSFD-Net	98.69	96.68	99.41	99.44	98.55
	RandLA-Net	97.58	98.27	99.39	99.35	98.65
Bridge 07	MSFD-Net	98.70	98.68	99.52	99.19	99.02
	RandLA-Net	98.80	94.83	99.50	99.84	98.24
Bridge 08	MSFD-Net	99.28	92.73	99.65	97.95	97.40
	RandLA-Net	95.92	94.71	99.28	62.63	88.14
Bridge 09	MSFD-Net	93.76	99.26	98.63	78.82	92.62
	RandLA-Net	92.44	98.96	98.60	79.18	92.29
Bridge 10	MSFD-Net	98.98	80.23	98.21	93.87	92.82
	RandLA-Net	98.24	91.30	98.33	86.29	93.54

Table A.2 compares the prediction results of the models in terms of Recall. MSFD-Net demonstrates recall values ranging from approximately 93% to 99%. This model exhibits higher recall in certain components, such as Abutment and Deck, across various bridges, indicating its ability to consistently capture these elements. However, there are fluctuations observed in recall values across different bridges and components, suggesting variability in its performance. In contrast, RandLA-Net displays recall values spanning around 62% to 99%. The model shows wider variability in recall across components and bridges compared to MSFD-Net. It notably struggles with lower recall, particularly evident in the Railing component across several bridges, indicating difficulties in accurately capturing this specific structural element.

Table A.3 compares the prediction results of the models in terms of F1 score. MSFD-Net consistently demonstrates robust F1 scores, in a range from approximately 93% to 98%. This model shows consistently in achieving higher F1 scores for components such as Abutment and Deck across different bridges. Its reliability in these areas signifies its ability to accurately identify and segment these structural elements. Conversely,

Table A.3: Comparative results of F1 score throughout the LOOCV test.

Sample	Model	Background	Deck	Abutment	Railing	Avg.
Bridge 01	MSFD-Net	97.96	96.89	98.05	91.81	96.18
	RandLA-Net	96.02	94.13	97.43	88.40	94.00
Bridge 02	MSFD-Net	97.47	97.46	96.89	86.05	94.47
	RandLA-Net	97.07	96.86	97.07	77.03	92.01
Bridge 03	MSFD-Net	98.11	97.13	97.96	89.70	95.72
	RandLA-Net	95.65	93.71	96.36	76.42	90.54
Bridge 04	MSFD-Net	98.13	97.97	97.70	94.07	96.97
	RandLA-Net	97.09	96.59	96.58	88.84	94.78
Bridge 05	MSFD-Net	97.19	96.42	96.49	87.20	94.32
	RandLA-Net	96.71	96.15	95.97	88.05	94.22
Bridge 06	MSFD-Net	98.57	97.25	98.21	89.33	95.84
	RandLA-Net	0.99	0.97	0.97	0.83	93.80
Bridge 07	MSFD-Net	99.21	98.07	97.98	97.17	98.11
	RandLA-Net	99.01	96.76	95.59	89.43	95.20
Bridge 08	MSFD-Net	97.10	96.08	98.65	94.03	96.47
	RandLA-Net	94.97	96.02	96.84	67.28	88.78
Bridge 09	MSFD-Net	96.28	96.38	95.98	85.15	93.45
	RandLA-Net	95.61	96.83	92.59	84.64	92.42
Bridge 10	MSFD-Net	94.25	88.58	97.11	93.02	93.24
	RandLA-Net	96.90	95.04	94.79	80.61	91.83

RandLA-Net displays F1 scores spanning a wider range, approximately from 67% to 95%. This model exhibits more significant variability in its F1 scores across components and bridges compared to MSFD-Net. Specifically, it faces challenges in achieving higher F1 scores, notably observed in components like Railing across multiple bridges.

This variability implies difficulties in consistently and accurately capturing these specific structural elements, potentially impacting its reliability in certain contexts.

Table A.4 illustrates the comparison of IoU scores between the models. MSFD-Net consistently demonstrates IoU scores within a range of approximately 89% to 98%. This model exhibits relatively higher IoU scores, particularly for components like Background and Abutment across different bridges. The consistency in achieving these scores suggests a robust capability to accurately delineate these structural elements. In contrast,

Table A.4: Comparative results of IoU throughout the LOOCV test.

Sample	Model	Background	Deck	Abutment	Railing	Avg.
Bridge 01	MSFD-Net	96.01	93.97	96.17	84.85	92.75
	RandLA-Net	92.35	88.92	95.00	79.21	88.87
Bridge 02	MSFD-Net	95.06	95.05	93.96	75.52	89.90
	RandLA-Net	94.31	93.92	94.32	62.65	86.30
Bridge 03	MSFD-Net	96.28	94.41	95.99	81.32	92.00
	RandLA-Net	91.67	88.16	92.97	61.84	83.66
Bridge 04	MSFD-Net	96.33	96.02	95.51	88.80	94.16
	RandLA-Net	94.35	93.40	93.40	79.93	90.27
Bridge 05	MSFD-Net	94.53	93.09	93.22	77.30	89.53
	RandLA-Net	93.64	92.59	92.25	78.65	89.28
Bridge 06	MSFD-Net	97.18	94.65	96.47	80.72	92.26
	RandLA-Net	97.17	94.40	94.11	70.31	89.00
Bridge 07	MSFD-Net	98.43	96.21	96.05	94.50	96.30
	RandLA-Net	98.04	93.72	91.56	80.87	91.05
Bridge 08	MSFD-Net	94.37	92.45	97.34	88.74	93.22
	RandLA-Net	90.41	92.34	93.88	50.70	81.83
Bridge 09	MSFD-Net	92.82	93.01	92.28	74.14	88.06
	RandLA-Net	91.59	93.86	86.20	73.37	86.25
Bridge 10	MSFD-Net	89.13	79.49	94.38	86.96	87.49
	RandLA-Net	93.99	90.55	90.10	67.51	85.54

RandLA-Net showcases IoU scores spanning approximately 51% to 95%. The model displays wider variability in IoU scores across components and bridges compared to MSFD-Net. Notably, it struggles to achieve higher IoU scores, particularly visible in components such as railing across multiple bridges, indicating challenges in accurately and consistently segmenting these specific elements.