LUKIS: A benchmark framework of geometric digital twinning for slab and beam-and-slab bridges

Ruodan Lu[1,2*]; Ioannis Brilakis[3]

* corresponding author

[1]School of Architecture, Building and Civil Engineering, Loughborough University, United Kingdom, Email: r.lu@lboro.ac.uk
[2]Darwin College, University of Cambridge, United Kingdom, Email: rl508@cam.ac.uk
[3]Department of Engineering, University of Cambridge, United Kingdom, Email: ib340@cam.ac.uk

Abstract

We devise, implement, and benchmark a framework LUKIS to automate the process of geometric digital twinning for existing slab and beam-and-slab bridges. LUKIS follows a top-down strategy to detect and twin bridge concrete elements in point clouds into an established data format Industry Foundation Classes. Existing software packages require modellers to spend many labour hours in generating shapes to fit point cloud sub-parts. Previous methods can generate surface primitives combined with rule-based classification to produce cuboid and cylinder models. While these methods work well in synthetic datasets or simplified cases, they encounter challenges when dealing with real-world point clouds. We tackle this challenge by investigating the entire workflow of geometric digital twinning for bridges and proposing LUKIS to auto-generate bridge objects without needing to generate low-level surface primitives. We implement LUKIS on a single software platform. Experiments demonstrate its ability to rapidly twin geometric bridge concrete elements. Compared to manual operation, LUKIS reduces the overall twinning time by at least 95.4% while the twinning quality (spatial accuracy) is improved. It is the first framework of its kind to achieve the geometric digital twinning for primary concrete elements of bridges on one platform. It lays foundations for researchers to generate semantically enriched digital twins.

Key words: Digital Twin, IFC, Point Cloud Data, Bridge Inspection

1 Introduction

Bridges are fundamental to a nation's transport infrastructure network. Every time a bridge collapse occurs, we are reawakened to the fact that our nation's most iconic structures need constant vigilance in the form of maintenance. In the United States (US) and the United Kingdom (UK), federal and state agencies, and highway authorities have developed bridge inspection and rating tools (FHWA, 2012; Flaig & Lark, 2000) aimed at prioritizing bridge rehabilitation projects. The US and the UK spend \$12.8 billion and £4 billion every year respectively to address deteriorating bridges and maintain their road network (ASCE, 2013; NAO, 2018). The reasons behind these enormous costs are mainly because the manual inspection data available in existing Bridge Management Systems (BMS) does not meet the standard of information needed for sound decision-making (ASCE, 2017). Innovative digital bridge documentation techniques are urgently needed to boost bridge management productivity.

Building Information Modelling (BIM) has been introduced and adopted for many decades in the Architecture, Engineering, and Construction sector (Sacks et al., 2018). However, BIM is not an appropriate model for real-time operational response. Its design-centric and static nature impedes its usage within the Operation & Maintenance (O&M) phase and lifecycle asset management. Statistics show that the percentage of using BIM technology on over half of the infrastructure projects in the US and Europe has doubled or even tripled from 2013 to 2017 (Buckley & Logan, 2017). However, the implementation concentrates in the design stage and is almost absent in the O&M phase (Table 1).

Table 1 Project stage at which BIM technology provide value (Buckley & Logan, 2017)

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

A Digital Twin (DT) (Grieves & Vickers, 2016; Parrott & Lane, 2017) comprises both three-dimensional (3D) geometry of the infrastructure components as well as a comprehensive set of semantic information, including material, functions, relationships between the components, and so on (Borrmann, et al., 2018b). It could be further enriched with other information, similar to what researchers implement on the Building Information Models, such as sensor data (Davila Delgado et al., 2016) and damage information (Hüthwohl et al., 2018). This is particularly useful for bridge inspection practice that is currently based on qualitative visual assessment and high-level comparisons, as the texture and damage data can be properly integrated with the geometry at the element-level of the virtual model of a bridge. The essential feature of DTs is their 3D geometry (Borrmann & Berkhahn, 2018) so that "geometric" is used to highlight the DT with only its geometry data, i.e. gDT. We can generate gDTs using the point clouds collected with laser scanners. Laser scanning is a technique to capture an object's exact size and shape into the computer world as a collection of XYZ co-ordinates. In the following texts, we introduce first (1) the *Exchange Information Requirements* (*EIRs*) of DTs; and then (2) provide a brief review of software solutions to see how far they have achieved in terms of the degree of automation on twinning from point clouds, according to the *EIRs*.

Exchange Information Requirements (EIRs)

The end-users of bridge DTs include but not limited to the inspectors, structural engineers, and decision makers. The *Exchange Information Requirements* (*EIRs*) (ISO, 2017) define the information that will be required by the end-users from both their own internal team and from suppliers/sub-contractors for the development of the project and for the operation of the completed built bridge. The *EIRs* should clearly articulate the information requirements for each supplier/sub-contractor and describe the expected information deliverables in terms of documents, model files, and structured information. However, the nature of the *EIRs* depends on the complexity of the project, the experience, and the requirements of the end-users (Ashworth et al., 2019). Experienced end-users may develop very detailed *EIRs*, whilst others may only set out high-level requirements and some basic rules, leaving the supplier/sub-contractors to propose how those requirements will be met (Ashworth et al., 2017; Kassem et al., 2016). Detailed *EIRs* of a DT are extensive and complex. Developing such thorough *EIRs* is outside the scope of this study. Given the dynamic and real-time updated nature of a DT, the following bullets outline the fundamental information that a bridge DT should contain for a specific moment. Broadly, *EIRs* should include:

- ***EIR 1*****:** Component-level digital representation. This means the geometric resolution of a bridge DT should at component-level, not assembly-level or even lower level (Sacks et al., 2017). Bridge component types include primary (e.g. pier, girder, pier cap, deck) and secondary (e.g. wing walls, handrails) elements (Kedar, 2016). Components that are totally occluded, or that are too small or invisible to be discerned due to insufficient scan resolution are not provided.
- ***EIR 2*****:** Component explicit geometry representation and property sets. This means a bridge DT is prepared for the use as an "Inspection Digital Model" in a BMS. A bridge DT should represent the as-is geometries of components (Borrmann & Berkhahn, 2018). Existing methods of documenting bridge data measure geometries accurate to 10 cm or 5 cm and round them up (Kedar, 2014).This is a rough estimation and notation. Defining the level of geometric detail (geometric spatial accuracy) of a bridge DT is beyond the scope of this research.
- ***EIR 3*****:** Component taxonomy. The components making up a bridge DT should not only be modelled but should also be identified and labelled with their component types (Koch & König, 2018; Ma et al., 2018).
- ***EIR 4*****:** Component implicit information such as spatial semantics of attributes and structural relationships, material, cost, schedule, and so on (Fink, 2018; Teizer & Melzner, 2018; Treeck et al., 2018). A bridge DT should be sufficiently semantically meaningful to provide most of the information needed for decision-making concerning the repair, retrofit or build of a bridge (Sacks, Kedar, et al., 2018).
- ***EIR 5*****:** Component damage information and the association to the bridge parts. Damage type (e.g. structural/ non-structural crack, spalling, scaling, efflorescence etc.), location, and orientation should be exactly identified and embedded into the bridge DT along with the texture/image data for each visible component (Hüthwohl et al., 2018).

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

- ***EIR 6*****:** All information requirements should be presented in a vendor-neutral data format, such as Industry Foundation Classes (IFC) (Borrmann et al., 2018b; Koch & König, 2018; Sacks et al., 2018).

There are already many capable software packages on the market such as Autodesk, Bentley, Trimble and ClearEdge3D, that provide the most advanced digital twinning solutions. For example, ClearEdge3D can automatically extract pipes in a plant point cloud as well as specific standard shapes like valves and flanges from industry catalogues followed by fitting built-in models to them through a few clicks and manual adjustments (ClearEdge3D, 2019). This means ClearEdge3D can achieve a certain degree of automation as the *EIRs 1, 2* and/or *3* can be partially automated. However, the spec-driven component library of ClearEdge3D can only recognize and fit point cloud subparts with standardised shapes based on an industry specification table. Limitations of state-of-the-art modelling software application have been highlighted in research (Agapaki et al., 2018). Modellers must first manually segment a point cloud into subparts, and then manually fit 3D shapes to them (*EIR 1* & *EIR 2*). Fitting accurate 3D shapes to the segmented point clusters is challenging because the set of allowable primitives is limited in most software applications (Wang et al., 2015) so that modellers must customize unusual shapes. Then, modellers need to enrich the resulting gDT with other explicit and implicit information, such as component's taxonomy (*EIR 3*), connectivity, aggregation and so on (*EIR 4*), and defects (*EIR 5*). Then, all *EIRs* need to be exported or converted in IFC format (*EIR 6*).

Real-world reinforced concrete (RC) bridge components contain skews and imperfections and cannot be simply fitted using regular generic shapes. This article investigated the entire manual twinning process for existing RC bridges from point clouds using CloudCompare and Autodesk Revit. Up until the end of the process, only *EIRs 1*, *2*, *3*, and *6* were partially satisfied. Although the Revit's Family Editor provides a powerful capability to create shapes in a free-form manner, almost all the total twinning time is spent on customizing shapes and fitting them to corresponding point clusters (note: the manual operation time and results will be presented in Section 4). The significant "bottlenecks" of the manual operation using software packages are listed as follow:

- Existing software packages can semi-automatically extract standardized shapes in point clouds. However, they cannot automatically extract non-canonical shapes, which are frequently present in RC bridges. Manual shape customization is laborious and time-consuming.

- *EIRs 1, 2, 3,* and *6* can only be manually achieved. *EIRs 4* and *5* are unavailable within existing applications.
- No single software can offer a one-stop gDT generation solution. Modellers need to shuttle intermediate results in different formats back and forth between different software packages during the twinning process, giving rise to the possibility of information loss.

To tackle abovementioned challenges in practice, we propose LUKIS (combines the authors' family names), a twinning framework for existing bridges, aiming to meet *EIRs 1*, *2*, *3*, and *6*, i.e. the *EIRs* required to generate a gDT with component semantic labels. *EIRs 4* and *5* are beyond the scope of this research. In the next section, we review existing research methods related to *EIRs 1*, *2*, *3* and *6*. We then outline LUKIS in Section 3 followed by demonstrating the experiments in Section 4. We discuss the results and draw our conclusions in Section 5.

2 Background

Previous research methods of twinning infrastructure using point clouds can be divided into two groups according to the *EIRs*: (1) object detection methods (*EIRs 1* and *3*); and (2) 3D solid model fitting methods (*EIRs 2* and *6*). We summarize each in turn in the following texts.

2.1 Object Detection in Point Clouds

Object detection aims to meet *EIRs 1 & 3.* In this work, "detection" is defined as the combination of *clustering* (from points to point clusters, i.e. *EIR 1*) and *classification* (labelling the point clusters, i.e. *EIR 3*). This problem has been intensively studied in research. Most existing clustering methods follow a "bottom-up" approach, which goes from points to surfaces followed by semantic labelling to derive objects (Dimitrov & Golparvar-Fard, 2015; Zhang et al., 2015). Most existing classification methods follow a "top-down" approach, which employs knowledge such as relationships and contexts to detect specific instances embedded in point clouds or to infer the semantics of components in a geometric model (Ahmed et al., 2014; Riveiro et al., 2016). We review both clustering and classification methods as well as investigate how far they have solved these challenges. Specific limitations are also identified.

2.1.1 Clustering methods

Existing clustering works usually take bottom-up approach to meet *EIR 1* (Zhang et al., 2014). As its name implies, the bottom-up approach pieces together low-level features like points to generate higher-level features successively. Higher-level features include but not limited to surface normals (Macher et al., 2017), meshes (Marton et al., 2009), surface patches (Zhang et al., 2015), non-uniform BSpline surfaces (NURBS) (Dimitrov et al., 2016), and voxels (Vo et al., 2015). Four main methods arise from the literature: RANdom Sample Consensus (RANSAC) (Zhang et al., 2015), Region Growing (Walsh et al., 2013), Hough-Transform (Díaz-Vilariño et al., 2015), and Octree paradigm (Xu et al., 2018). Whilst the RANSAC algorithm is effective to detect planar surfaces in the presence of outliers, it suffers from spurious-planes, which are frequently produced around the boundaries (Jung et al., 2014). In addition, RANSAC requires prior knowledge about the data, meaning that the selection of a fixed number of shape hypotheses implies that a prior estimate of the inlier ratio is available (Schnabel et al., 2007). This is often not the case in the practice of twinning for real-world bridges. Region Growing (RG) is also a widely used scheme for point cloud clustering whereas it has the boundary weakness issue. It excels when the point cloud does not suffer from substantive occlusions. However, it tends to over segments objects when non-trivial occlusions are present (Dimitrov & Golparvar-Fard, 2015). Hough-Transform (HT) is another commonly used clustering method. Adan & Huber (2011) proposed HT methods to detect walls in building point clouds. HT becomes computationally prohibitive when the number of dimension increases. For example, it requires a 5D parameter space for cylinder detection (Rabbani, 2006). Thus, HT is sensitive to parameter dimensions and cannot be directly applied to shapes characterized by too many parameters. This constraint impedes its use in the detection of bridge objects, which often contain skews and imperfections, and cannot be described using generic shapes with limited parameters. Octree-Based (OB) methods have been proposed to tackle the issue of computational complexity (Su et al., 2016). However, the segmentation accuracy of OB methods is sensitive to the voxel size. Vo et al. (2015) proposed an OB algorithm which can semi-automatically adjust the voxel size using an adaptive octree. However, it faces the difficulty of patch clustering for low point density regions.

2.1.2 Classification methods

Top-down methods are usually used to meet *EIR 3*, because above-mentioned bottom-up detection schemes are rarely suitable for point cloud classification (Riveiro et al., 2016). Without high-level information, it could be difficult to determine whether low-level primitives such as local surfaces or voxels belong to the same instance. The top-down approach usually combines a set of engineering criteria and classifies objects in point clouds that meet the criteria. Top-down classification methods are robust because domain-specific information such as known parameters (Ahmed et al., 2014), object instances (Dore & Murphy, 2014), and spatial relationships (Koppula et al., 2011), are invariant to factors such as pose and appearance. Perez-Gallardo et al. (2017) suggested a semantic model based system to detect four object classes in an industrial scene using topological information. Riveiro et al. (2016) used topological constraints to segment masonry bridge point cloud through normal clustering. However, this algorithm largely depends on data quality so that it is difficult to generalize it to adapt real data suffering from occlusions and non-uniformly distributed points. Ma et al., (2017) leveraged relationship knowledge and shape features to classify bridge 3D solid objects (Figure 1). However, the input of this method needs to be a geometric bridge model (not a point cloud), meaning that it assumes both *EIRs 1* and *2* are already achieved.

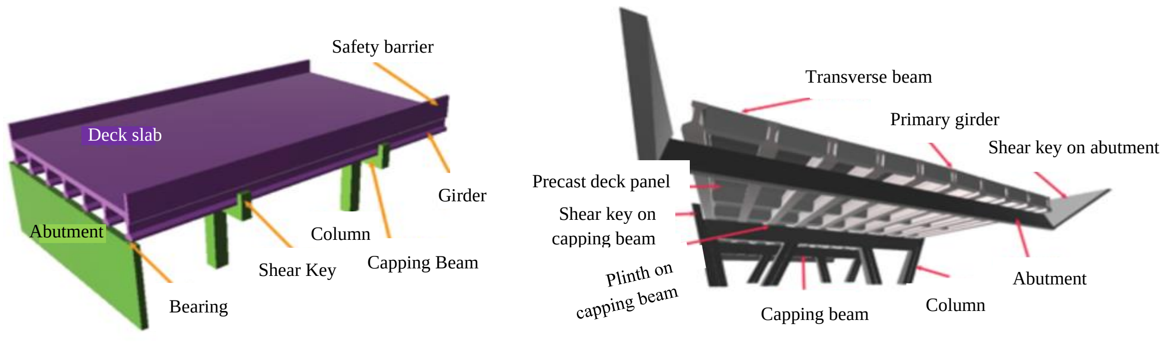


Figure 1 Bridge Object classification (EIR 3) (adapted from Ma et al., 2017)

2.1.3 Learning-based methods

Learning-based methods are seeing a lot of use recently to detect objects in point clouds, aiming to satisfy both *EIRs 1* & *3* at once (Xiong et al., 2013). Maturana & Scherer (2015) proposed a supervised 3D Convolutional Neural Network called *VoxNet* to classify objects using voxelized data. Instead of transforming a point cloud into 3D voxel grids, Qi et al. (2016) introduced a deep neural network called *PointNet*, which can directly consume points. These methods often require a substantial down-sampling procedure before they can be fed even in high performance computing systems such as TensorFlow. However, the resulting compressed datasets often lose feature information along the way.

2.2 Model Fitting to Point Clusters

2.2.1 Fitting methods

Model fitting aims to meet *EIR 2* in the context of twinning for infrastructure. In other words, model fitting aims to use computer graphic techniques to form the 3D shape of a point cluster. Existing shape representation methods are categorized into four groups: Implicit Representation (Schnabel et al., 2007), Boundary Representation (B-Rep) (Oesau et al., 2014), Constructive Solid Geometry (CSG) (Xiao & Furukawa, 2014), and Swept Solid Representation (SSR) (Ochmann et al., 2016). Implicit Representation is of limited usefulness when twinning real-world bridge components because only a very limited number of shapes can be represented by algebraic formulations. Although B-Rep is the most popular representation in computer graphics, the presence of significant occlusions in point clouds challenges the forming of closed mesh models. CSG methods have been proposed whereas only elementary primitives such as cuboids, cylinders, and so on are used to constitute the built environment. For example, Zhang et al. (2014) designed a classifier to classify infrastructure components (*EIR 3*) and fit them with 3D shapes (*EIRs 1 & 2*). However, it only used cuboids and cylinders to fit pier, girder, and deck in synthetic bridge point clouds (Figure 2). Laefer & Truong-Hong (2017) used a SSR method to extrude the cross-sections of steel beams in point clouds, assuming the standardized profiles do not have any damages or deformation.

Table ***2*** summarizes the most related research methods of the gDT generation from point clouds in terms of the application area and the IFC maturity.

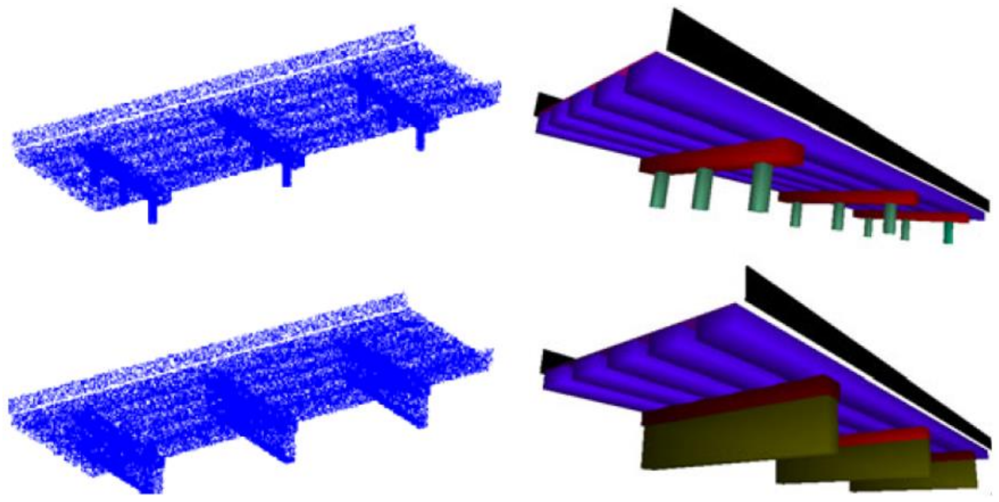


Figure 2 Fitted cuboids and cylinders to synthetic bridge point clusters (EIR 2) (Zhang et al., 2014)

2.2.2 IFC Geometric Representations

A gDT making up of multiple fitted geometric shapes need to be represented in IFC format so that it can be exchanged and adopted by various software tools (*EIR 6*) (Venugopal et al., 2012). IFC geometry representations can be grouped into four classes (Borrmann et al., 2018a): Bounding Boxes, Curves, Surface models, and Solid models. More specific, Bounding Boxes are usually used as placeholders for digital twinning. They can be represented using *IfcBoundingBox*. *IfcCurve* and its subclasses can be used to model line objects. Freeform curved edges (i.e. splines) and curved surfaces are required to model complex geometries. *IfcTriangulatedFaceSet* can be used to represent polygons with an arbitrary number of edges, or triangular mesh. *IfcBSplineSurface* can be used for representing curved surfaces such as NURBS. Finally, *IfcSweptAreaSolid* and its subclasses can be used to present extruded solids. How to choose an IFC geometric representation depends on the specific application scenario where we use the object being twinned. Whilst software programs only need low-level triangulated mesh to visualize the geometry, gDTs authoring tools usually use high-level B-Rep- or CSG-based objects so that the geometries could be edited. Details of the IFC geometric representation can be found in (Borrmann et al., 2018a).

Table 2 State of research on gDT generation from point clouds in terms of IFC maturity

Area		**Surfaces**	**Labelled Surfaces**	**Shapes without Labels**	**Labelled Clusters**	**Labelled 3D Model**	**Enriched IFC Model**
	Bridge		(Riveiro et al., 2016) (Zhang et al., 2015)	(Walsh et al., 2013)	(Lu et al., 2018)	(Zhang et al., 2014) (Lu & Brilakis, 2019)	*(Hüthwohl et al., 2018) *(Ma et al., 2018)
	Building	(Dimitrov et al., 2016) (Vo et al., 2015) (Jung et al., 2014) (Truong-Hong et al., 2013) (Marton et al., 2009)	(Xu et al., 2018) (Wang et al., 2015) (Díaz-Vilariño et al., 2015) (Xiong et al., 2013) (Adan & Huber, 2011) (Koppula et al., 2011)	(Dimitrov & Golparvar-Fard, 2015) (Xiao & Furukawa, 2014) (Schnabel et al., 2007)	(Qi et al., 2016) (Armeni et al., 2016) (Su et al., 2016)	(Macher et al., 2017) (Ochmann et al., 2016) (Valero, Adán, & Bosché, 2016) (Valero et al., 2012)	*(Belsky et al., 2014)
	Industry				(Patil et al., 2017) (Su et al., 2016) (Son et al., 2015) (Ahmed et al., 2014) (Son et al., 2013)	(Laefer & Truong-Hong, 2017)	

*Input: gDT

IFC Maturity

3 Proposed Framework: LUKIS

3.1 Overview

We intend to solve the problem of twinning for existing RC bridges while improving our understanding of the specific challenges and the extent to which we have solved the problem. The main objective of this research is to develop a benchmark framework that paves the foundations for future research to build upon.

The top-down approach is based on domain-knowledge and relies on a symbolic description of the simplified world. Its goal is to use a set of pre-defined recognition criteria or rules to mimic human intelligence, hence suffers from some inherent limitations. Inevitably, a combinatorial explosion of the number of rules occurs due to the complexity of the environment and it is impossible to predict all situations (especially unknown ones) that will be encountered by an autonomous entity. Yet, the limitations of the top-down approach are irrelevant in the context of slab and beam-and-slab RC bridges. This is because the level of variance of these types of bridges is relatively low compared to other real-world objects. Bridge components are distinct 3D solid objects, and their taxonomies for any given context are finite and well defined. Therefore, this article proposes a novel top-down framework called LUKIS, which exploits bridge engineering knowledge as guidance to directly extract labelled point clusters corresponding to bridge components and then to reconstruct them into IFC objects.

Real-world bridges are neither perfectly straight nor flat. Bridge geometries are defined by horizontal curved alignments, vertical elevations, and varying cross-sections. A slicing-based algorithm is proposed to tackle these difficulties. The algorithm is repeatedly used throughout the whole framework until all the components are detected and modelled. The algorithm can deal with the skew complexity and can quickly select a set of candidate locations for target objects. The global topology of a bridge can also be well approximated using multiple slices. The general thrust behind the slice-based representation is the Cavalieri's principle (Kern & Bland, 1948), which serves as the theoretical guidance of LUKIS. Given an arbitrary 3D solid along the X-axis, extending from $x = a$ to $x = b$ (Figure 3), the solid is divided into n equally thick slices and define the usual partition: $\Delta x = \frac{b-a}{n}$, and $x_i = a + i\Delta x$, for $i = 1,2,\dots,n$, then Cavalieri's principle states that:

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

where $A(x)$ is the area of the cross-section of a slice.

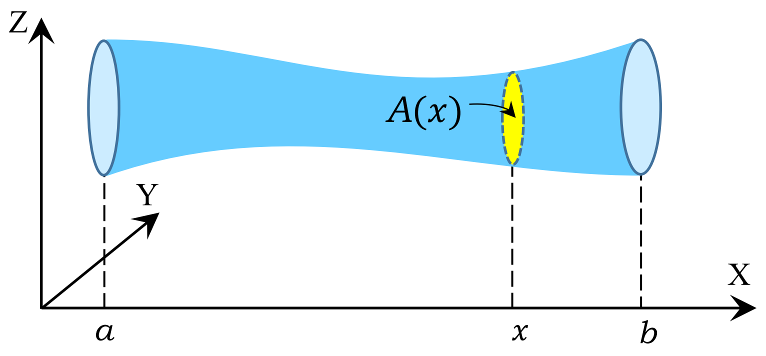


Figure 3 An arbitrary 3D shape represented using its cross-section A(x) along X-axis

LUKIS only twins four most important and highly detectable primary components of typical RC slab and beam-and-slab bridges: slab, pier, pier cap, and girder. It consists of two successive processes (Figure 4): **Process 1.** Detection of the four bridge component types in point clouds, aiming to achieve *EIRs 1* & *3*, and **Process 2.** Run-time model fitting to the point clusters using IFC standards, aiming to achieve *EIRs 2* & *6*. Specifically, LUKIS starts with a registered point cloud of an RC bridge. Irrelevant points such as vegetation, trees, traffic and so on are manually removed. It then aligns the cropped point cloud using Principal Analysis Component (PCA) such that its centre axis i.e. horizontal alignment, is roughly parallel to the X-axis of the global coordinate system. This step outputs a roughly aligned bridge point cloud as none of bridges could be positioned exactly parallel to the axes due to their real-world skewed geometry. Next, LUKIS proposes a four-step object detection method (Process 1) to detect the four components underlying in a cropped bridge point cloud in the form of labelled point clusters. A manual refinement procedure is then conducted to remove the noise remaining from Process 1. Next, LUKIS proposes an IFC object fitting method (Process 2) to fit the refined labelled point clusters with 3D IFC objects. The output of Process 2 i.e. the final output of LUKIS, is an IFC file of a bridge gDT. The novelty of LUKIS lies in the fact that it integrates Process 1 and 2 into a single framework that provides a semi-automated method to twin four bridge components from point clouds at one go. We outline each Process in the following sections.

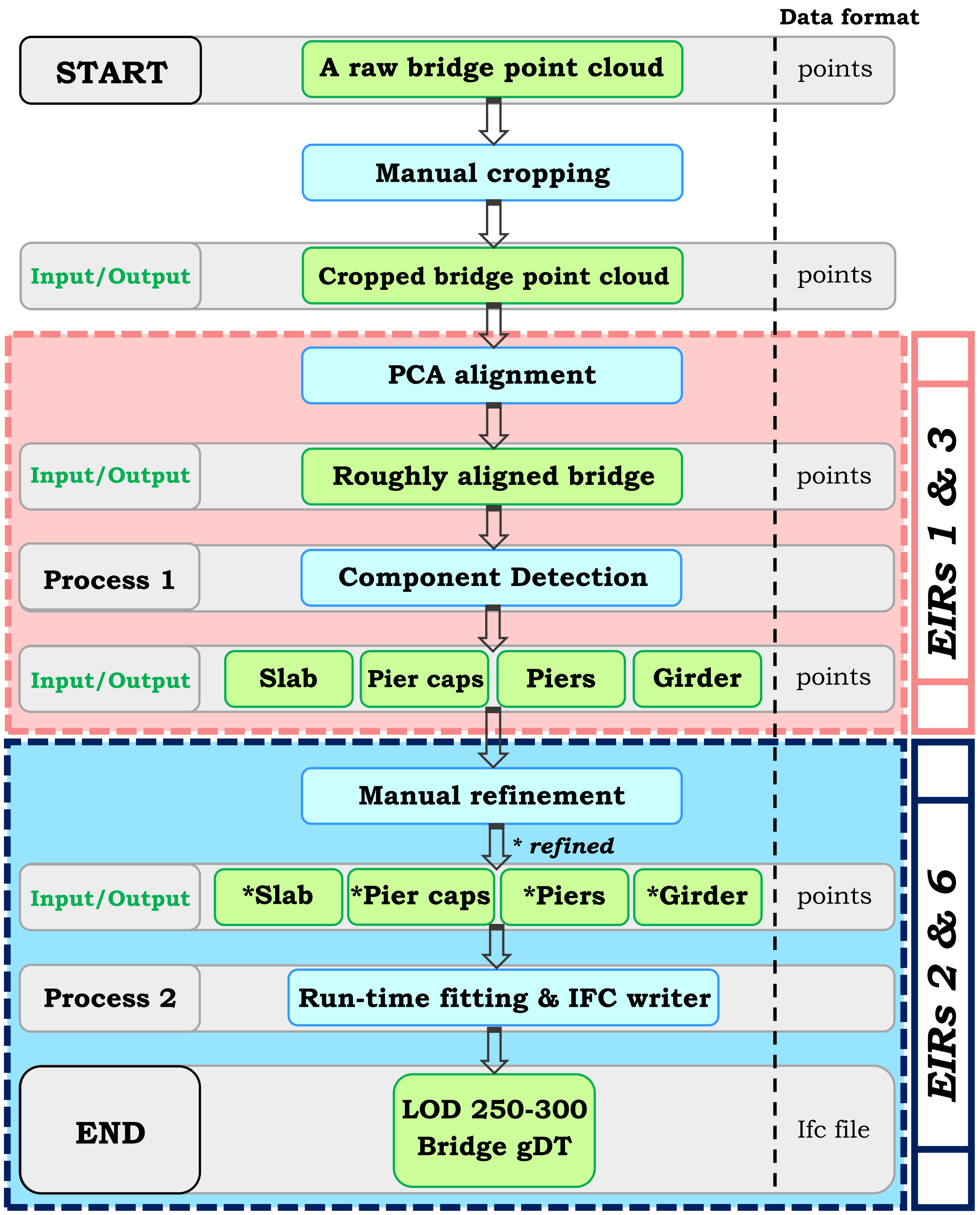
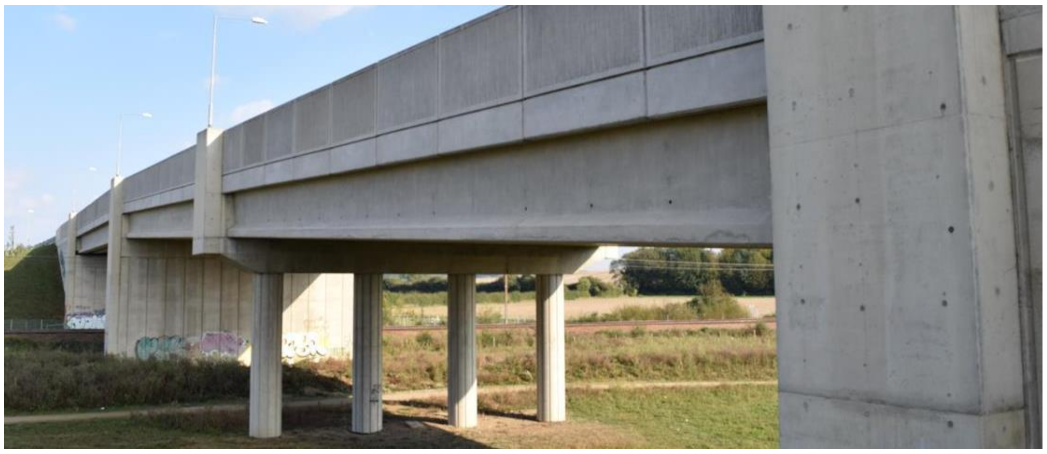


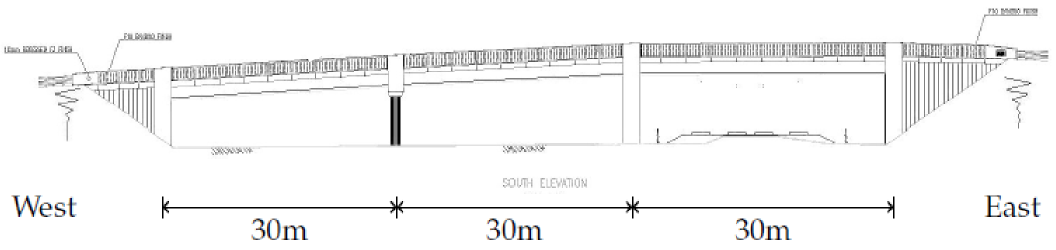
Figure 4 The workflow of the proposed framework LUKIS

3.2 Process 1: Primary Component Detection for *EIRs 1 & 3*

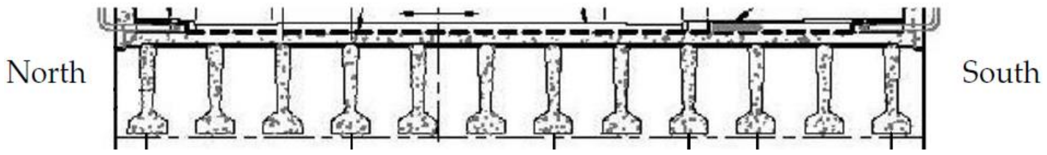
We use the point cloud of the Nine Wells Bridge to demonstrate the development of Process 1. The bridge was constructed to the south west of Addenbrooke's Hospital in Cambridge in 2009 to carry a new road over a railway line. It comprises three spans of approximately 30 m in length, each constructed from twelve precast concrete "SY beams" with an in-situ concrete deck slab. The bridge also contains a group of four cylindrical piers with a pier cap on top, a wall-like pier, and two wall-like vertical abutments as supports. The side view shows its varying elevations over the three spans (Figure 5). Process 1 bypasses the stage of surface generation altogether and directly obtains labelled point clusters. It breaks down a large bridge point cloud into sub-datasets through a recursive slicing algorithm. That is, the method slices the point cloud by means of a "virtual parallel scalpel" with a specified equal thickness. This algorithm is repeatedly used with sub-datasets until target objects are found and all small detection problems are solved. Step 1 segments a whole aligned bridge point cloud into two classes: pier assembly α_m and deck assembly. This is achieved by segmenting the bridge point clouds into multiple slices along the X-axis and comparing the height feature of each slice using a discriminative parameter ρ_1 (Figure 6). Then, the search area is shrunk to the pier assembly α_m. Step 2 aims to detect pier area in pier assembly α_m using a discriminative parameter ρ_2. Each extracted pier assembly α_m from Step 1 is considered as a miniature of an entire bridge point cloud so that Step 2 follows the same strategy as Step 1. Next, Step 3 further narrows down the search area and detects pier caps in a pier area by investigating the normals. Finally, Step 4 aims to detect girders. We segment the entire deck assembly cluster into several spans followed by using the density histograms to detect girders in each span. All over-segments are merged, and we finally acquire the four labelled point clusters of bridge components (*EIRs 1 & 3*). Details of Process 1 and the four-step object detection method can be found in (Lu et al., 2018).



(a)



(b)



(c)

Figure 5 (a) Northwest view; (b) Side view; and (c) SY beams of the Nine Wells Bridge (Graham, 2014)

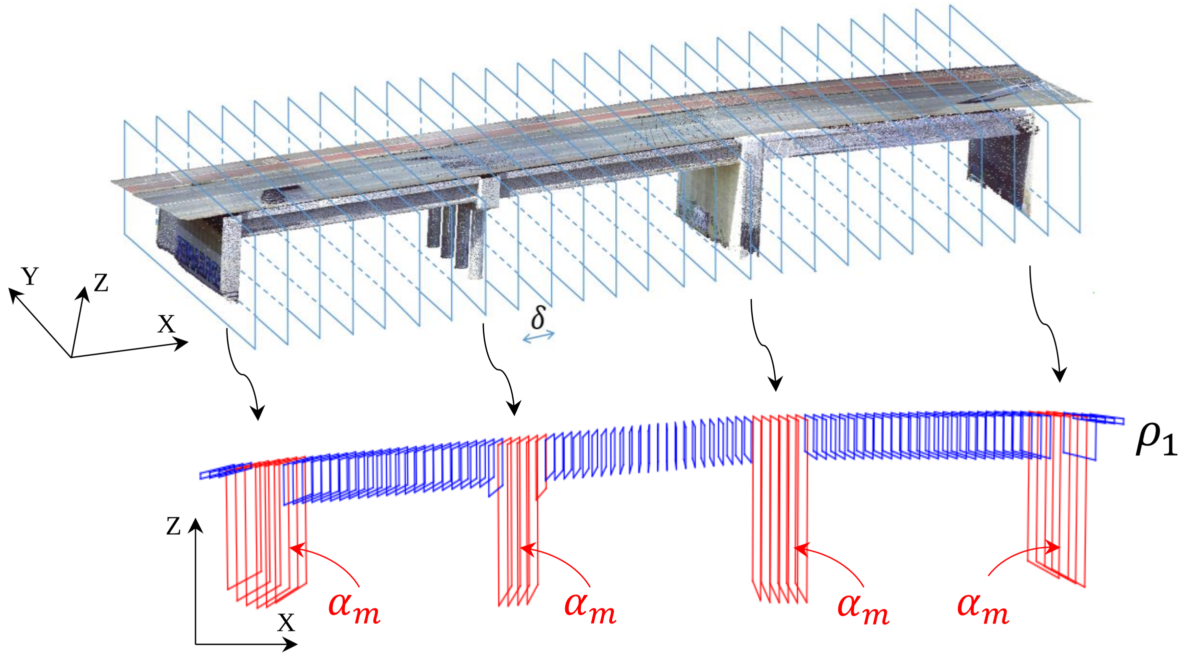


Figure 6 Slicing along X-axis in Step 1 of Process 1

3.3 Process 2: IFC object fitting for *EIRs 2 & 6*

Process 2 suggests a novel object fitting method that can twin an RC bridge into IFC format using the four types of point cluster. The inputs of Process 2 are the point clusters generated from Process 1. The output is an IFC file corresponding to level of detail LOD 250 – 300. The four types of component are represented with detailed geometries through multiple slice models. SSR is preferred wherever possible if the cross-section of each slice is deemed constant. Process 2 consists of two major steps: Step 1, geometric feature extraction of point clusters; and Step 2, *IfcObjects* fitting to the extracted features. We use the Model View Definition proposed by Sacks et al. (2018), which provides a binding to the IFC4 Add2 standard for exchanging bridge DTs and defines the mandatory IFC entities in a bridge inspection scenario.

For a slab point cluster *IfcSlab*, we use a similar but not identical slicing scheme to that proposed in Process 1 to slice the deck slab into multiple slices. The slicing does not take a parallel pattern but is rather oriented along the normal direction of the curved alignment of the slab (Figure 7 (a)). The problem of twinning the entire slab is transformed into twinning slab slices by assuming each one of them is straight along its tangent direction and the cross-section of each slice is constant. That is to say, the horizontal alignment is assumed gap-freedom and can be approximated by concatenating many alignment segments. We use a 2D profile *IfcArbitraryClosedProfileDef* and *IfcExtrudedAreaSolid* to describe the cross-section of each extruded slice, which is defined by a 2D *ConvexHull* α-shape. Pier cap point clusters *IfcBeam* are also represented as Swept Solid using the outline of the α-shape on the XY-plane. Then, for pier point clusters *IfcColumn*, we use a fuzzy-logic algorithm to first classify the cross-section shape into *circular*, *quadrilateral*, and *others*. A *cylindrical* pier is represented as Swept Solid using a 2D *IfcCircleProfileDef*, while a *quadrilateral* and *other* shape pier are represented using multiple slice models through *IfcArbitraryClosedProfileDef* and the *IfcExtrudedAreaSolid* (Figure 7 (b)). Finally, for girder point clusters *IfcBeam*, we use a template matching method to find the best-match girder type in existing beam catalogue. We finally acquire a bridge gDT in IFC format, making up of four types of component (*EIRs 2* & *6*). Details of Process 2 and the proposed fitting method can be found in (Lu & Brilakis, 2019).

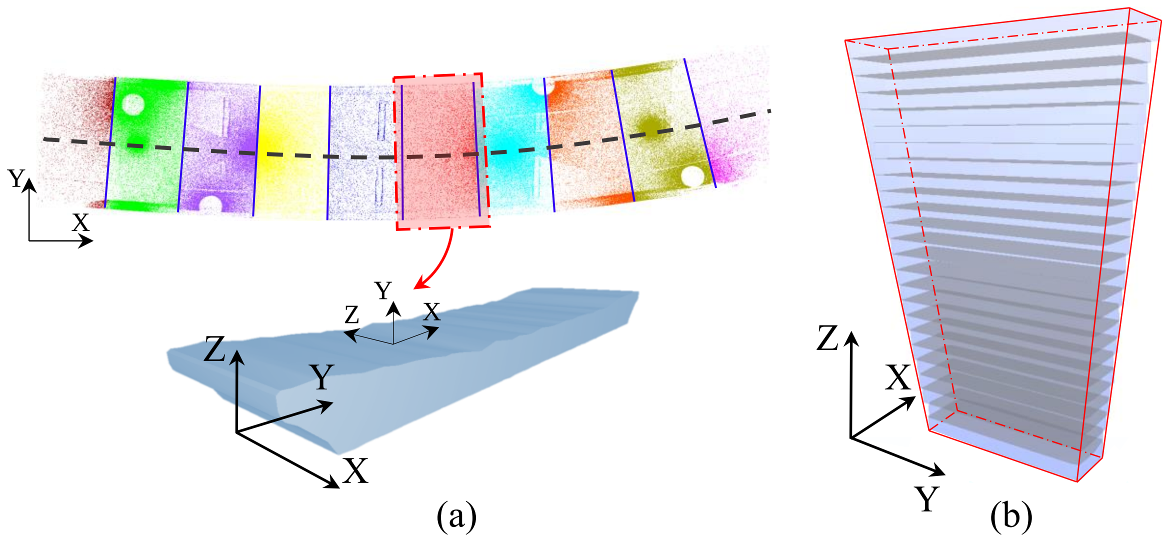


Figure 7 Slicing and twinning of Process 2 for (a) slab and (b) pier

4 Research Activities

4.1 Implementation

Ten highway RC slab and beam-and-slab bridge point clouds were used to test the proposed framework (

Figure *8*). Data can be downloaded from http://doi.org/10.5281/zenodo.1233844. Detailed statistics of the data can be found in (Lu et al., 2018).

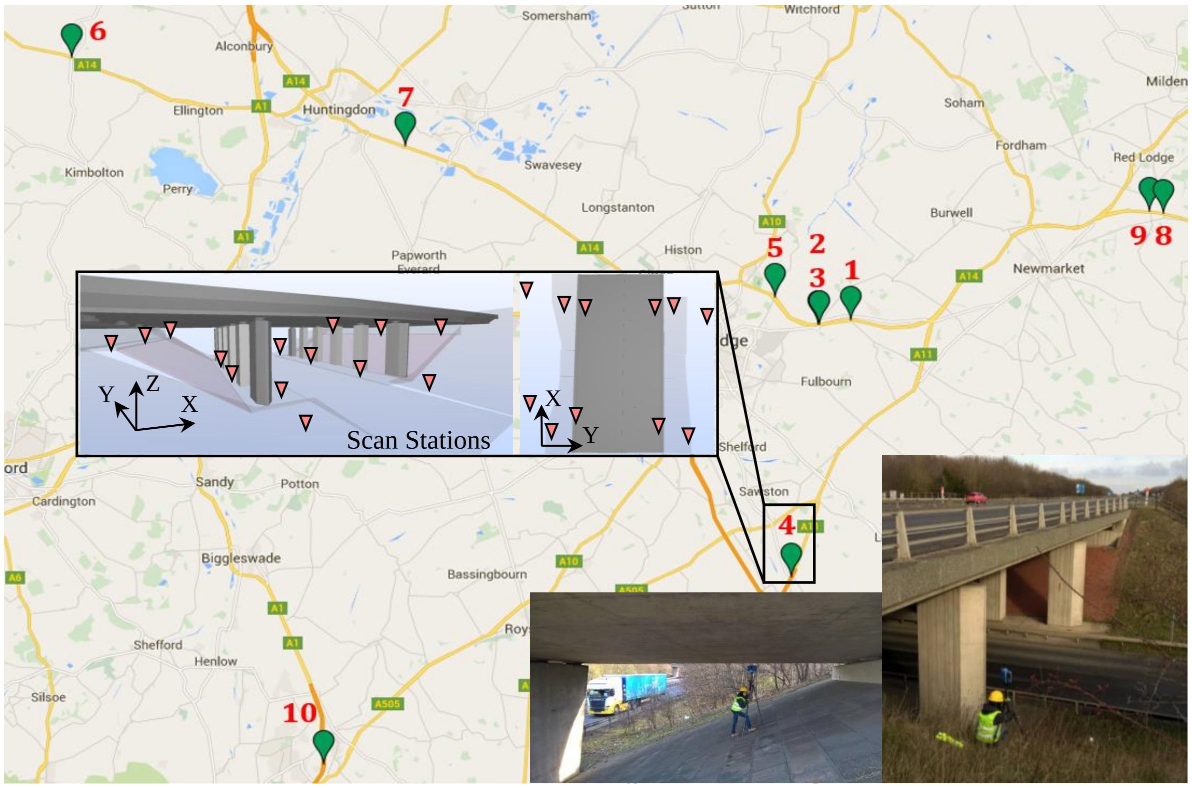


Figure 8 Locations of the ten bridges, scan stations, and the on-site activities

We implemented LUKIS on Gygax (Hüthwohl et al., 2017) into a proof of concept prototype software on a desktop computer with the following system configurations : CPU Intel Core i7-4790K 4.00GHz, Memory 32GB, SSD 500GB. Gygax contains three projects: GygaxCore (C#), GygaxVisu (C#), and PclWrapper (C++). GygaxCore defines basic functionalities such as the data structures and interfaces of this platform. GygaxVisu supports visualization. Helix Toolkit was used to visualize all supported data sources in a single 3D space. PclWrapper allows Gygax to access Point Cloud Library API in a C# environment. IFC Engine DLL was used to read and visualize IFC files on a logical and geometrical level.

The raw bridge point clouds were extremely noisy because we collected the data with live traffic. Data cleaning-up is needed before running any experiments. We developed a user-defined 2D clipping polygon function on Gygax to manually remove irrelevant points such as the on-site traffic, vegetation, ground surface and so on (Figure 9). To do so, the current graphics view of 3D points was mapped onto a 2D screen by controlling the viewport to conduct the clipping. The 3D points in the world coordinates (x_w, y_w, z_w) were transformed into a camera space followed by perspective projection and affine transformation to convert the camera coordinates into the screen coordinates (x_{pix}, y_{pix}) using:

$$p(x_{pix}, y_{pix}) = \frac{1}{Z_c} K[\mathrm{I} \quad 0][R|T] \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}, \qquad \text{Eq.2}$$

where z_c is the world coordinate z_w in the camera system, K is the intrinsic parameter of a camera, and R and T are the extrinsic parameters. We then used PCA to align the cropped bridge point cloud followed by implementing Process 1 i.e. bridge component detection, and Process 2 i.e. IFC object fitting, on Gygax as two successive modules. Details of Process 1 can be found in (Lu et al., 2018) and details of Process 2 can be found in (Lu & Brilakis, 2019).

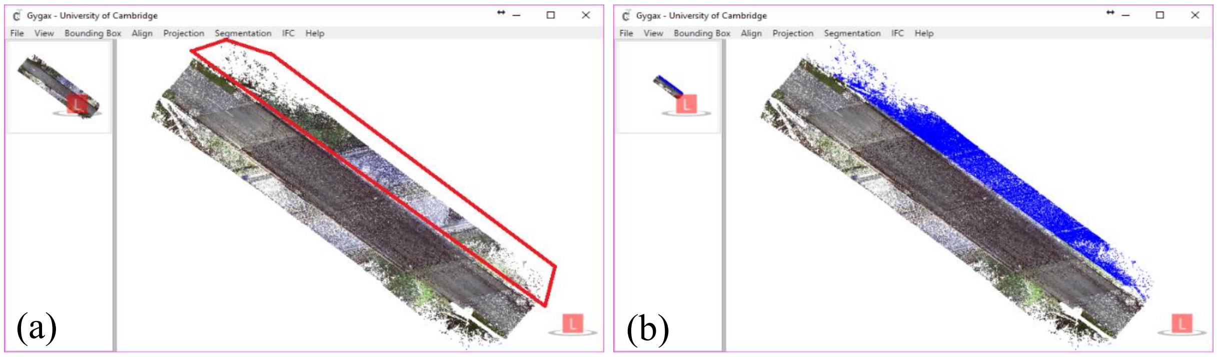

Gygax - University of Cambridge
File View Bounding Box Align Projection Segmentation IFC Help
(a)
Gygax - University of Cambridge
File View Bounding Box Align Projection Segmentation IFC Help
(b)

Figure 9 Top view of Bridge 1 (a) clipping polygons; (b) selected points are coloured and removed

4.2 Manual Operation

For Process 1, three ground-truth (GT) datasets: *GT A*, *GT B*, and *GT C*, were manually created to conduct Step 1, Step 2, and the entire object detection method, respectively by researchers who are familiar with the construction/form of the slab and beam-and-slab bridges (Figure 10). The GT datasets are optimal desired outputs to compare against those generated from LUKIS. We recorded the cleaning-up time (T1) and the object detection time (T2) for the manual operation as well as for LUKIS. Then, for Process 2, a set of 3D geometric models, i.e. *GT D* was manually created to compare against the resulting LOD 250 – 300 gDTs. Again, we recorded the manual refinement time (T3) and the twinning time (T4) for the manual operation as well as for LUKIS. The manual operation time, as well as the twinning time of LUKIS, are given in the next section.

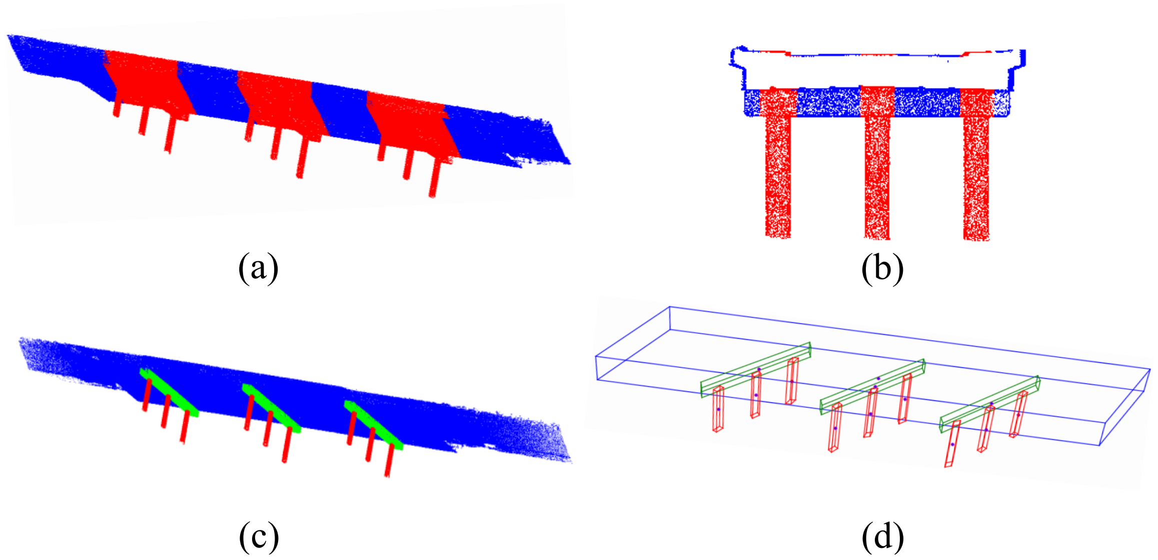
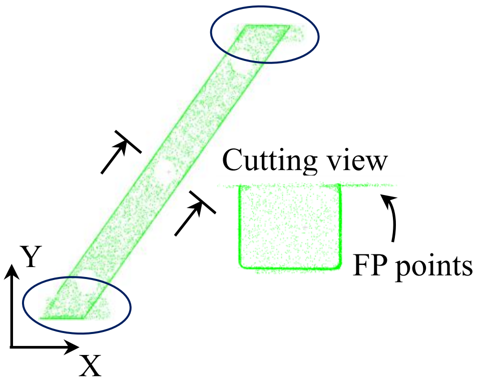


***Figure 10** Process 1 (a) GT A of Step 1; (b) GT B of Step 2; (c) GT C point clusters of the entire method; (d) GT C oriented bounding boxes of the point clusters (example: Bridge 1)*

4.3 Experiments

The experiments of LUKIS start by determining the two key discriminative parameters ρ_1 and ρ_2 in Process 1. We statistically estimated them by comparing against *GT A* and *GT B* using point-wise performance metrics Precision (Pr), Recall (R), and F1-score (F1). We conducted a grid-search over the value space (0, 1) and computed the empirical receiver operating characteristic (ROC). The optimal values of ρ_1 and ρ_2, i.e. ρ_1^* and ρ_2^*, were identified when the distance to the perfect classification in the ROC was minimized. Then, we evaluated the entire method with the other parameters deduced based on ρ_1^* and ρ_2^*. For each bridge, the evaluation was conducted using both bounding-box-wise and point-wise metrics (Table 3). The average Pr, R and F1 of bounding-box-wise component detection for all ten bridges were 100%, 98.5%, and 99.2%. For point-wise evaluation, the micro-average scores of Pr/R/F1 was 98.4% (±3.1%) for the ten bridges. The false discovery rate (FDR) revealed that some boundary points between adjacent point clusters were detected as false positives (FPs), e.g. *Bridge 1* $\text{FDR}_{\text{pierCap}}$ 8.6%, and *Bridge 7* FDR_{pier} 2.75% (detailed results can be found in Lu et al., 2018). Although the number of these FP points was limited, the effectiveness of Process 2 would be affected due to the incorrectly generated concave hulls. Thus, the FPs in each point cluster generated from Process 1 should be removed before implementing Process 2. To do so, we used the developed cropping function presented earlier (see Figure 9) to manually select the FP points in each point cluster followed by removing them (Figure 11).



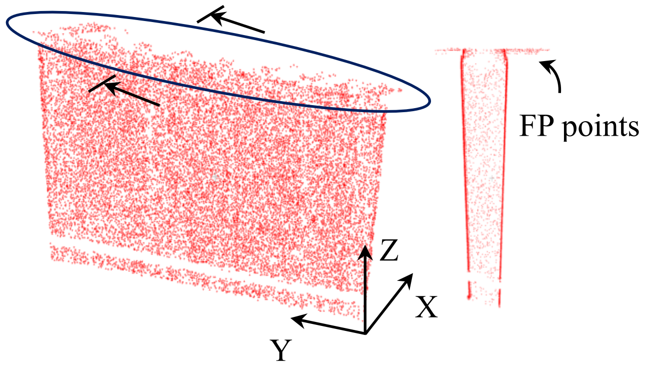


Figure 11 FP points around boundaries: Bridge 1 PierCap 2 (L); Bridge 7 Pier (R)

LUKIS then used the refined point clusters as inputs to twin the bridges in IFC format. The twinning quality of both the manually generated models i.e. *GT D*, and the ones generated from LUKIS were evaluated and compared using distance-based cloud-to-cloud (C2C) metrics. To do so, we converted the *GT D* and the automated models from LUKIS in IFC format into point clouds. Then, we compared the LUKIS point clouds (denoted *LUKIS*) and the GT point clouds (denoted *GT*) of each bridge against its original point cloud (denoted *Real*) respectively. In total, six out of ten bridges were modelled better by using LUKIS than by the manual operation. There are some challenges scenarios, e.g. *Bridge 7* and *Bridge 10*, whose C2C values were significant, raising the overall $\overline{\mathrm{C2C}}$. The overall $\overline{\mathrm{C2C}}_{LUKIS}$ of the other eight bridge gDTs was 5.6 (±1.7) cm while the $\overline{\mathrm{C2C}}_{GT}$ was 7.0 (±2.1) cm (detailed results can be found in Lu & Brilakis, 2019).

Table 3 Summarized performance of LUKIS for EIRs 1, 2, 3 & 6

	Bounding-box-wise (%)			Point-wise (%)	$\overline{C2C}$ (cm)	
	$\overline{P}_r$	$\overline{R}$	$\overline{F}_1$	micro-average $\overline{\mathrm{P}}_\mathrm{r}/\overline{\mathrm{R}}/\overline{\mathrm{F}_1}$		
EIRs 1 & 3	100	98.5	99.2	98.4	--	
EIRs 2 & 6	--			--	10 Bridges	8 Bridges (w/o *Bridge 7&10*)
					7.05	5.6

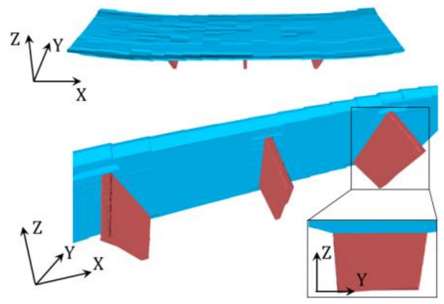
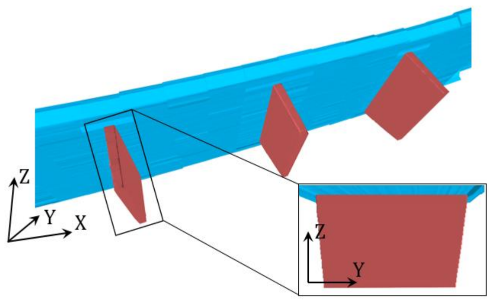
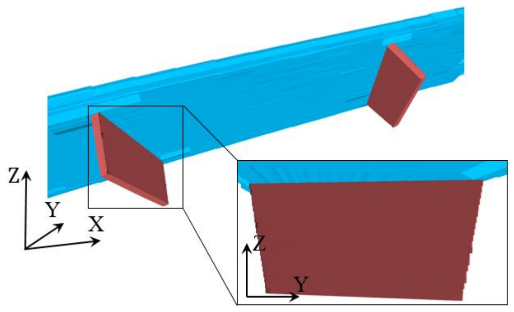
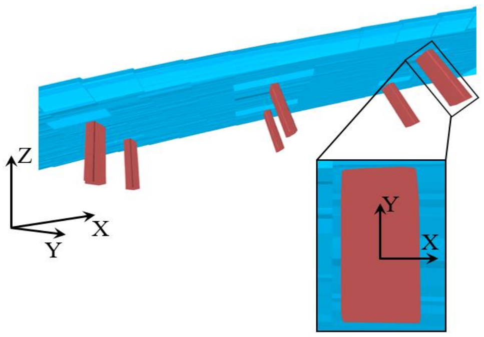
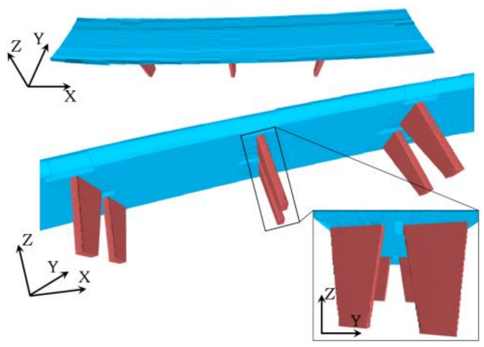
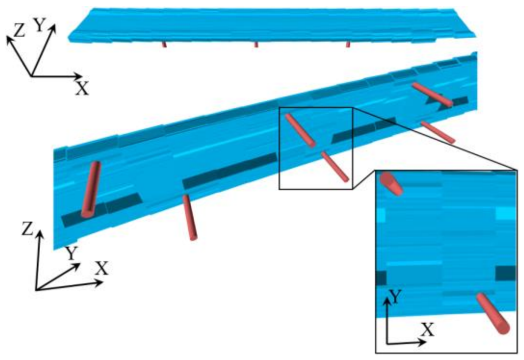
The time spent on data cleaning-up for both the manual operation and for LUKIS were almost the same (Table 4), i.e. $\overline{\mathrm{T1}}_{\mathrm{manual}}$=58.1 (±6.9) minutes, $\overline{\mathrm{T1}}_{\mathrm{LUKIS}}$=52.0 (±11.7) minutes. Likewise, the time spent on object detection for both the manual operation and for LUKIS were in the same order of magnitude, i.e. $\overline{\mathrm{T2}}_{\mathrm{manual}}$=12.0 (±7.6) minutes, $\overline{\mathrm{T2}}_{\mathrm{LUKIS}}$=8.3 (±0.8) minutes. Then, the time spent on manual refinement (T3) for the manual operation and for LUKIS was 9.0 (±1.7) minutes and 7.8 (±1.6) minutes, respectively. The most time-consuming phase in the whole gDT generation is the twinning phase, i.e. from point clusters to 3D shapes. We spent in average $\overline{\mathrm{T4}}_{\mathrm{manual}}$=27.6 (±8.2) hours (≈1656 minutes) to manually twin a gDT from labelled point clusters.

The average twinning time of LUKIS was 37.8 (±14.2) seconds, which was trivial compared to the manual operation ($\overline{\mathrm{T4}}_{\mathrm{manual}}$ = 27.6 hours). Table 4 summarizes the time breakdown of the manual operation and that of the LUKIS. As shown, 95.4% of the overall manual operation time was spent on twinning, whereas this was the most rapid task in LUKIS (less than 1% time of LUKIS). LUKIS only took a fraction of the manual twinning time (4%) to complete the entire workflow. Table 5 illustrates part of the resulting gDTs of the ten bridges as well as their twinning time T4.

Table 4 Time breakdown of the manual operation and LUKIS

(minutes)	Cleaning-up $\overline{T1}$	Object Detection $\overline{T2}$	Refinement $\overline{T3}$	Twinning $\overline{T4}$	Overall
Manual	58.1	12.0	9.0	1656	1735.1
LUKIS	52.0	8.3	7.8	0.63	68.7

Table 5 LOD 250 – 300 gDTs generated from LUKIS

	Bridge 2	*Bridge 3*
gDT		
T4(s)	25.3	23.7
	Bridge 5	*Bridge 6*
gDT		
T4(s)	23.5	46.3
	Bridge 8	*Bridge 10*
gDT		
T4(s)	39.8	65.5

5 Conclusions

This article presented LUKIS, an automated benchmarking framework to generate gDTs for existing highway RC bridges from point clouds. This work focuses on the compilation of the bridge gDTs while reducing the time and effort required to acquire them. LUKIS was developed into a software prototype using the coding platform Gygax, which allows practitioners to generate gDTs through a one-single human-computer interface instead of switching between different software applications using different data formats. The framework consists of two main processes: Process 1 component detection, aiming to detect four bridge component types in the form of labelled point clusters; and Process 2 run-time IFC object fitting, aiming to generate the bridge gDT using the four types of point clusters. We implemented LUKIS on the largest published bridge point cloud database, consisting of ten highway slab and beam-and-slab RC bridges. In Process 1, the number of misclassified points was limited. This demonstrates the robust performance of LUKIS on component detection in real-world point clouds featuring defects and how top-down reasoning could facilitate the detection. The limited FP points were present at the boundaries of each point cluster and were suggested to be manually removed in Gygax. These refined labelled point clusters served as input of Process 2, where LUKIS successfully generated bridge gDTs, using the four types of point clusters. Distance-based C2C metrics were used to evaluate the twining quality. The overall $\overline{\mathrm{C2C}}_{LUKIS}$ was 5.6 cm with a standard deviation 1.7 cm, if we did not count the two non-trivial C2C values (i.e. *Bridge 7* and *Bridge 10*), meaning that an improvement of 20% was made. Abnormal values exist mainly due to major occlusions rather than twinning deviations. LUKIS also exhibits difficulties in twinning complex superstructure geometry. Although imperfections exist, experimental results demonstrate that the performance of LUKIS is consistent and less liable to human error. *EIRs 1*, *2*, *3*, and *6* have been largely achieved. In addition, LUKIS made an enormous timesaving, significantly outperforming the manual practice. The time of cleaning-up and refinement for the manual process was 67.1 minutes, while LUKIS reduced it by 11%. The object detection and refinement time of LUKIS was 31% and 13.3% less than that of the manual operation, respectively. In total, LUKIS reduces the overall manual twinning time by 96%. Assuming that LUKIS takes as much time as the manual operation on cleaning-up, object detection, and refinement, i.e. $\overline{\mathrm{T1}}_{\mathrm{LUKIS}}=\overline{\mathrm{T1}}_{\mathrm{manual}}$, $\overline{\mathrm{T2}}_{\mathrm{LUKIS}}=\overline{\mathrm{T2}}_{\mathrm{manual}}$, $\overline{\mathrm{T3}}_{\mathrm{LUKIS}}=\overline{\mathrm{T3}}_{\mathrm{manual}}$, then, it still can realize a direct timesaving of 95.4%. This means LUKIS can automate the most time-consuming twinning step, significantly overriding the current practice.

Current version of LUKIS can only twin four most important bridge components of two most representative RC bridge types. In the future, (1) researchers will investigate more bridge types and components with various configurations. LUKIS has the potential to deal with geometries that are more complex by integrating additional technique layers in each step. Then, it tends to generate undulating-surface and jagged-edge models, which might be over-detailed for end-users. So, (2) researchers will investigate how and to what extent to smooth unnecessary undulations and jagged edges. Next, the approximate alignment tends to generate gaps and overlaps between adjacent slice models of slab. Thus, (3) researchers will also investigate a method that can generate a gap-less single curved slab model by sweeping along a 2D alignment curve using *IfcSectionedSolidHorizontal* or along an arbitrary 3D curve using *IfcFixedReferenceSweptAreaSolid*. Last, LUKIS is not yet fully automatic. Several steps, such as cleaning-up and refinement still require human intervention. Approximate 87% time of LUKIS was spent on these two manual steps. (4) Future work will focus on automating the remaining manual work, especially the refinement step, by implementing noise-filtering algorithms. Overall, LUKIS, together with other works in this field, is the start of a longer effort to create the toolsets needed to build the national DT roadmap. It lays solid foundations for future research of digital twinning for existing infrastructure.

Acknowledgements

This research work is supported by EPSRC, Infravation SeeBridge project under Grant Number No. 31109806.0007, and Cambridge Trimble Fund. We would like to thank for their supports. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of EPSRC, Infravation SeeBridge, or Trimble.

References

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

Agapaki, E., Miatt, G., & Brilakis, I. (2018). Prioritizing object types for modelling existing industrial facilities. *Automation in Construction*. https://doi.org/10.1016/j.autcon.2018.09.011

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

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

ASCE. (2013). 2013 Report Card for America's Infrastructure, Bridges. Availble at http://2013.infrastructurereportcard.org/, accessed 3 April, 2019. *ASCE*.

ASCE. (2017). 2017 Report Card for America's Infrastructure, Bridges. Available at https://www.infrastructurereportcard.org/cat-item/bridges/, accessed 3 April, 2019. *ASCE*.

Ashworth, S., Tucker, M., & Druhmann, C. (2017). Employer's Information Requirements (EIR): A BIM case study to meet client and facility manager needs. *EUROFM'S 16th Research Symposium*.

Ashworth, S., Tucker, M., & Druhmann, C. K. (2019). Critical success factors for facility management employer's information requirements (EIR) for BIM. *Facilities*. https://doi.org/10.1108/F-02-2018-0027

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

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

Borrmann, A., & Berkhahn, V. (2018). Principles of Geometric Modeling. In A. Borrmann, M. König, C. Koch, & J. Beetz (Eds.), *Building Information Modeling* (pp. 27–41). https://doi.org/10.1007/978-3-319-92862-3_2

Borrmann, A., König, M., Koch, C., & Beetz, J. (2018b). Building Information Modeling: Why? What? How? In *Building Information Modeling* (pp. 1–24). https://doi.org/10.1007/978-3-319-92862-3_1

Buckley, B., & Logan, K. (2017). The Business Value of BIM for Infrastructure 2017. Available at https://www2.deloitte.com/content/dam/Deloitte/us/Documents/finance/us-fas-bim-infrastructure.pdf, accessed 3 April, 2019. *Dodge Data & Analytics*, 1–68.

ClearEdge3D. (2019). *Structure Modelling Tools. Available at https://www.clearedge3d.com/,* accessed 16 June 2019.

Davila Delgado, J. M., Butler, L. J., Gibbons, N., Brilakis, I., Elshafie, M. Z. E. B., & Middleton, C. (2016). Management of structural monitoring data of bridges using BIM. *Proceedings of the Institution of Civil Engineers - Bridge Engineering*, 1–15.

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

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

Dimitrov, A., Gu, R., & Golparvar-Fard, M. (2016). Non-Uniform B-Spline Surface Fitting from Unordered 3D Point Clouds for As-Built Modeling. *Computer-Aided Civil and Infrastructure Engineering*, *31*(7), 483–498. https://doi.org/10.1111/mice.12192

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

FHWA - Federal Highway Administration. (2012). *Estimated 2012 Costs to Replace or Rehabilitate Structurally Deficient Bridges. Available at https://www.fhwa.dot.gov/bridge/nbi/sd2012.cfm,* accessed 3 April, 2019.

Fink, T. (2018). BIM for Structural Engineering. In *Building Information Modeling* (pp. 329–336). https://doi.org/10.1007/978-3-319-92862-3_19

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

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

Grieves, M., & Vickers, J. (2016). Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems. In *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches*. https://doi.org/10.1007/978-3-319-38756-7_4

Hüthwohl, P., Armeni, I., Fathi, H., & Brilakis, I. (2017). *Gygax construction IT research platform for 2D & 3D*. Retrieved from https://github.com/ph463/Gygax

Hüthwohl, P., Brilakis, I., Borrmann, A., & Sacks, R. (2018). Integrating RC Bridge Defect Information into BIM Models. *Journal of Computing in Civil Engineering*, *32*(3), 04018013. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000744

International Organization for Standardization (ISO). (2017). ISO/DIS 19650-1 - Part 1: Concepts and Principles. *ISO 2017*.

Jung, J., Hong, S., Jeong, S., Kim, S., Cho, H., Hong, S., & Heo, J. (2014). Productive modeling for development of as-built BIM of existing indoor structures. *Automation in Construction*, *42*, 68–77. https://doi.org/10.1016/j.autcon.2014.02.021

Kassem, M., Dawood, N., Chahrour, R., Vukovic, V., Ahmad, A. M., & Hafeez, M. A. (2016). Principles and recommendations for client information requirements for BIM enabled construction projects in Qatar. *International Journal of Product Lifecycle Management*. https://doi.org/10.1504/ijplm.2016.10001531

Kedar, A. (2014). Guide for Documenting Bridges and Road Structures Data. *Kedmor Engineers Ltd*.

Kedar, A. (2016). *SeeBridge Project Document: Criteria for evaluation of complete SeeBridge System*.

Kern, W. F., & Bland, J. R. (1948). Spherical segment. *Solid Mensuration With Proofs.2nd Ed.New York, NY: Wiley*.

Koch, C., & König, M. (2018). Data Modeling. In *Building Information Modeling* (pp. 43–62). https://doi.org/10.1007/978-3-319-92862-3_3

Koppula, H. S., Anand, A., Joachims, T., & Saxena, A. (2011). Semantic Labeling of 3D Point Clouds for Indoor Scenes. *Neural Information Processing Systems*, 1–9.

Laefer, D. F., & Truong-Hong, L. (2017). Toward automatic generation of 3D steel structures for building information modelling. *Automation in Construction*, *74*, 66–77. https://doi.org/10.1016/j.autcon.2016.11.011

Lu, R. (2019). *Automated Generation of Geometric Digital Twins of Existing Reinforced Concrete Bridges. Doctoral t*. https://doi.org/10.17863/CAM.36680

Lu, R., & Brilakis, I. (2019). Digital twinning of existing reinforced concrete bridges from labelled point clusters. *Automation in Construction*. https://doi.org/10.1016/j.autcon.2019.102837

Lu, R., Brilakis, I., & Middleton, C. R. (2018). Detection of Structural Components in Point Clouds of Existing RC Bridges. *Computer-Aided Civil and Infrastructure Engineering*. https://doi.org/10.1111/mice.12407

Ma, L., Sacks, R., Kattel, U., & Bloch, T. (2018). 3D Object Classification Using Geometric Features and Pairwise Relationships. *Computer-Aided Civil and Infrastructure Engineering*, *33*(2), 152–164. https://doi.org/10.1111/mice.12336

Macher, H., Landes, T., & Grussenmeyer, P. (2017). From Point Clouds to Building Information Models: 3D Semi-Automatic Reconstruction of Indoors of Existing Buildings. *Applied Sciences*, *7*(10), 1030. https://doi.org/10.3390/app7101030

Marton, Z. C., Rusu, R. B., & Beetz, M. (2009). On fast surface reconstruction methods for large and noisy point clouds. *2009 IEEE International Conference on Robotics and Automation*, *269*(1–2), 3218–3223. https://doi.org/10.1109/ROBOT.2009.5152628

Maturana, D., & Scherer, S. (2015). VoxNet: A 3D Convolutional Neural Network for real-time object recognition. *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 922–928. https://doi.org/10.1109/IROS.2015.7353481

NAO. (2018). A Short Guide to the Department for Transport. *National Audit Office*. Retrieved from https://www.nao.org.uk/wp-content/uploads/2018/02/A-Short-Guide-to-the-Department-for-Transport-2017.pdf

Ochmann, S., Vock, R., Wessel, R., & Klein, R. (2016). Automatic reconstruction of parametric building models from indoor point clouds. *Computers & Graphics*, *54*, 94–103. https://doi.org/10.1016/j.cag.2015.07.008

Oesau, S., Lafarge, F., & Alliez, P. (2014). Indoor scene reconstruction using feature sensitive primitive extraction and graph-cut. *ISPRS Journal of Photogrammetry and Remote Sensing*, *90*, 68–82. https://doi.org/10.1016/j.isprsjprs.2014.02.004

Parrott, A., & Lane, W. (2017). Industry 4.0 and the digital twin. Available at https://www2.deloitte.com/insights/us/en/focus/industry-4-0/digital-twin-technology-smart-factory.html, accessed 14 October, 2019. *Deloitte University Press*.

Patil, A. K., Holi, P., Lee, S. K., & Chai, Y. H. (2017). An adaptive approach for the reconstruction and modeling of as-built 3D pipelines from point clouds. *Automation in Construction*, *75*, 65–78. https://doi.org/10.1016/j.autcon.2016.12.002

Perez-Gallardo, Y., Cuadrado, J. L. L., Crespo, Á. G., & de Jesús, C. G. (2017). GEODIM: A Semantic Model-Based System for 3D Recognition of Industrial Scenes. In *Intelligent Systems Reference Library* (pp. 137–159). https://doi.org/10.1007/978-3-319-51905-0_7

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2016). PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *CVPR 2017*. Retrieved from http://arxiv.org/abs/1612.00593

Rabbani, T. (2006). Automatic Reconstruction of Industrial Installations Using Point Clouds and Images. Available at http://resolver.tudelft.nl/uuid:0012068e-93b4-4bd9-a9b3-9c579ae7c91a, accessed 3 April, 2019. *Publications on Geodesy*, *62*(May), 7401–7410.

Riveiro, B., DeJong, M. J., & Conde, B. (2016). Automated processing of large point clouds for structural health monitoring of masonry arch bridges. *Automation in Construction*, *72*(3), 258–268. https://doi.org/10.1016/j.autcon.2016.02.009

Sacks, R., Eastman, C., Lee, G., & Teicholz, P. (2018). BIM Handbook. In *BIM Handbook*. https://doi.org/10.1002/9781119287568

Sacks, R., Kedar, A., Borrmann, A., Ma, L., Brilakis, I., Hüthwohl, P., … Muhic, S. (2018). SeeBridge as next generation bridge inspection: Overview, Information Delivery Manual and Model View Definition. *Automation in Construction*, *90*, 134–145. https://doi.org/10.1016/j.autcon.2018.02.033

Sacks, R., Ma, L., Yosef, R., Borrmann, A., Daum, S., & Kattel, U. (2017). Semantic Enrichment for Building Information Modeling: Procedure for Compiling Inference Rules and Operators for Complex Geometry. *Journal of Computing in Civil Engineering*. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000705

Schnabel, R., Wahl, R., & Klein, R. (2007). Efficient RANSAC for Point-Cloud Shape Detection. *Computer Graphics Forum*, *26*(2), 214–226. https://doi.org/10.1111/j.1467-8659.2007.01016.x

Son, H., Kim, C., & Kim, C. (2013). KNOWLEDGE-BASED APPROACH FOR 3D RECONSTRUCTION OF AS-BUILT INDUSTRIAL PLANT MODELS FROM LASER-SCAN DATA. *ISARC*. https://doi.org/10.22260/ISARC2013/0096

Son, Hyojoo, Kim, C., & Kim, C. (2015). 3D reconstruction of as-built industrial instrumentation models from laser-scan data and a 3D CAD database based on prior knowledge. *Automation in Construction*, *49*, 193–200. https://doi.org/10.1016/j.autcon.2014.08.007

Su, Y., Bethel, J., & Hu, S. (2016). Octree-based segmentation for terrestrial LiDAR point cloud data in industrial applications. *ISPRS Journal of Photogrammetry and Remote Sensing*, *113*, 59–74. https://doi.org/10.1016/j.isprsjprs.2016.01.001

Teizer, J., & Melzner, J. (2018). BIM for Construction Safety and Health. In *Building Information Modeling* (pp. 349–365). https://doi.org/10.1007/978-3-319-92862-3_21

Treeck, C. van, Wimmer, R., & Maile, T. (2018). BIM for Energy Analysis. In *Building Information Modeling* (pp. 337–347). https://doi.org/10.1007/978-3-319-92862-3_20

Truong-Hong, L., Laefer, D. F., Hinks, T., & Carr, H. (2013). Combining an Angle Criterion with Voxelization and the Flying Voxel Method in Reconstructing Building Models from LiDAR Data. *Computer-Aided Civil and Infrastructure Engineering*, *28*(2), 112–129. https://doi.org/10.1111/j.1467-8667.2012.00761.x

Valero, E., Adán, A., & Bosché, F. (2016). Semantic 3D Reconstruction of Furnished Interiors Using Laser Scanning and RFID Technology. *Journal of Computing in Civil Engineering*, *30*(4), 04015053. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000525

Valero, E., Adán, A., & Cerrada, C. (2012). Automatic Method for Building Indoor Boundary Models from Dense Point Clouds Collected by Laser Scanners. *Sensors*, *12*(12), 16099–16115. https://doi.org/10.3390/s121216099

Venugopal, M., Eastman, C. M., Sacks, R., & Teizer, J. (2012). Semantics of model views for information exchanges using the industry foundation class schema. *Advanced Engineering Informatics*, *26*(2), 411–428. https://doi.org/10.1016/j.aei.2012.01.005

Vo, A.-V., Truong-Hong, L., Laefer, D. F., & Bertolotto, M. (2015). Octree-based region growing for point cloud segmentation. *ISPRS Journal of Photogrammetry and Remote Sensing*, *104*, 88–100. https://doi.org/10.1016/j.isprsjprs.2015.01.011

Walsh, S. B., Borello, D. J., Guldur, B., & Hajjar, J. F. (2013). Data Processing of Point Clouds for Object Detection for Structural Engineering Applications. *Computer-Aided Civil and Infrastructure Engineering*, *28*(7), 495–508. https://doi.org/10.1111/mice.12016

Wang, C., Cho, Y. K., & Kim, C. (2015). Automatic BIM component extraction from point clouds of existing buildings for sustainability applications. *Automation in Construction*, *56*, 1–13. https://doi.org/10.1016/j.autcon.2015.04.001

Xiao, J., & Furukawa, Y. (2014). Reconstructing the World's Museums. *International Journal of Computer Vision*, *110*(3), 243–258. https://doi.org/10.1007/s11263-014-0711-y

Xiong, X., Adan, A., Akinci, B., & Huber, D. (2013). Automatic creation of semantically rich 3D building models from laser scanner data. *Automation in Construction*, *31*, 325–337. https://doi.org/10.1016/j.autcon.2012.10.006

Xu, Y., Tuttas, S., Hoegner, L., & Stilla, U. (2018). Voxel-based segmentation of 3D point clouds from construction sites using a probabilistic connectivity model. *Pattern Recognition Letters*, *102*, 67–74. https://doi.org/10.1016/j.patrec.2017.12.016

Zhang, G., Vela, P. A., & Brilakis, I. (2014). Automatic Generation of As-Built Geometric Civil Infrastructure Models from Point Cloud Data. *Computing in Civil and Building Engineering (2014)*, 406–413. https://doi.org/10.1061/9780784413616.051

Zhang, Guangcong, Vela, P. A., Karasev, P., & Brilakis, I. (2015). A Sparsity-Inducing Optimization-Based Algorithm for Planar Patches Extraction from Noisy Point-Cloud Data. *Computer-Aided Civil and Infrastructure Engineering*, *30*(2), 85–102. https://doi.org/10.1111/mice.12063