Digital twinning of existing bridges from labelled point clusters

Digital Twinning of Existing Bridges from Labelled Point Clusters

Ruodan Lu[1] and Ioannis Brilakis[2]

Department of Engineering, University of Cambridge, United Kingdom
E-mail: [1]rl508@cam.ac.uk, [2]ib340@cam.ac.uk

Abstract –

The automation of digital twinning for existing bridges from point clouds has yet been solved. Whilst current methods can automatically detect bridge objects in points clouds in the form of labelled point clusters, the fitting of accurate 3D shapes to detected point clusters remains human dependent to a great extent. 95% of the total manual modelling time is spent on customizing shapes and fitting them to right locations. The challenges exhibited in the fitting step are due to the irregular geometries of existing bridges. Existing methods can fit geometric primitives such as cuboids and cylinders to point clusters, assuming bridges are made up of generic shapes. However, the produced geometric digital twins are too ideal to depict the real geometry of bridges. In addition, none of existing methods have evaluated the resulting models in terms of spatial accuracy with quantitative measurements. We tackle these challenges by delivering a slicing-based object fitting method that can generate the geometric digital twin of an existing reinforced concrete bridge from labelled point clusters. The accuracy of the generated models is gauged using distance-based metrics. Experiments on ten bridge point clouds indicate that the method achieves an average modelling distance smaller than that of the manual one (7.05 cm vs. 7.69 cm) (value included all challenging cases), and an average twinning time of 37.8 seconds. Compared to the laborious manual practice, this is much faster to twin bridge concrete elements.

Keywords –

Digital Twin, BrIM, BIM, Point Cloud Data, IFC

1 Introduction

The United States (US) and United Kingdom (UK) spend a lot of money every year ($12.8 billion and £4 billion, respectively) to address deteriorating bridges and maintain their road networks. The reasons behind these massive costs are in part because bridge owners face a major challenge with structuring and managing the data needed for rapid maintenance and retrofit of their assets. The data available in Bridge Management Systems (BMS) does not meet the standard of information needed for sound decision-making. There is a need for at least 315,000 bridge inspections per annum across the US and the UK, given the typical two-year inspection cycle [1] [2]. Visual inspection is still the most common form of condition monitoring. The resulting condition information from visual assessment is then entered into a BMS, such as the AASHTOWare (US) or the NATS (UK). However, these BMSs are geared primarily to make system-wide prioritization decisions based on high-level comparisons of condition data. They do not assess the actual condition of a particular bridge component and of a particular location of the component. Having a Geometric Digital Twin (gDT) would be quite useful for this purpose as texture and damage information can then be properly integrated with the geometry at the component-level of the virtual 3D representation of a bridge. However, bridge owners today do not create such gDTs for existing bridges in the maintenance stage [3]. The following text reviews the current practice of digital twinning using point clouds, i.e. the process to acquire a gDT for an existing asset. This explains why the gDT implementation is so limited.

Major vendors such as Autodesk, Bentley, Trimble and ClearEdge3D, etc. provide the most advanced digital twinning software 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. This means ClearEdge3D can realize a certain degree of automation. However, ClearEdge3D can only recognize and fit point cloud subparts with standardised shapes such as rectangular walls, pipes, valves, flanges, and steel beams, based on an industry specification table. Other commercial applications do not automate object detection nor arbitrary shape fitting. Fitting accurate 3D shapes to individual point clusters is challenging because the set of allowable primitives is limited in most software applications. Reinforced concrete (RC) bridge components usually have complicated shapes,

containing skews and imperfections, and cannot be simply fitted using idealized generic shapes. Modellers must manually create an accurate solid form to fit each point cluster as none of the existing software packages can do this automatically. Although modelling software such as Revit provides fine flexibilities that allow users to design a shape in a freeform manner (via Revit's Family editor) (Figure 1), 95% of the total modelling time is spent on customizing shapes and fitting them to point clusters [4].

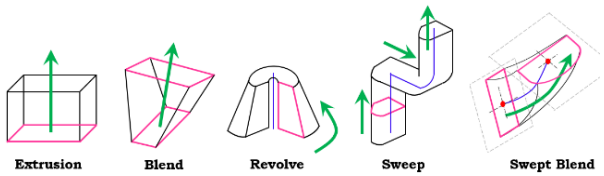


Figure 1. Forms available in Revit Family editor

2 Research Background

Unlike building geometries which are defined in a grid system, real-world bridge geometries are more complex, which are defined with curved alignments, vertical elevations, and varying cross-sections. Extensive manual effort is required for practitioners to manually customize 3D accurate models to fit the underlying bridge components in arbitrary shapes in point clouds. We define "fitting" in this context as leveraging computer graphic techniques to form the 3D shape of a point cluster, a subpart of a point cloud. The 3D shape is approximate in the sense that it describes the geometry or the shape of a point cluster to produce its digital 3D representation to an acceptable quality based on the specific required level of detail.

2.1 Fitting techniques

There is no universal solution to describe an object. How to choose a representation mainly depends on (1) the nature of the object being modelled, (2) the particular modelling technique that we choose to use, and (3) the application scenario where we bring the object to life. Existing shape representation methods can be categorized into four groups: Implicit Representation, Boundary Representation, Constructive Solid Geometry, and Swept Solid Representation. We review each of these in the following texts.

2.1.1 Implicit Representation

One solid modelling approach is based on the representation of 3D shapes using mathematical formulations, i.e. implicit functions. Common implicit surface definitions include, but are not limited to:

Table 1 Common implicit surfaces

Shape	Equation
Plane	$ax + by + cz = d$
Sphere	$x^2 + y^2 + z^2 = r^2$
Ellipsoid	$(\frac{x}{r_x})^2 + (\frac{y}{r_y})^2 + (\frac{z}{r_z})^2 = 1$

Given that only a very limited number of primitives can be represented exactly by algebraic formulations, implicit functions are of limited usefulness when modelling bridge objects. There is a trade-off between the accuracy of the representation and the bulk of information used for shapes that cannot be represented by mathematical formulations. We present three other basic modelling types: Boundary Representation (B-Rep), Constructive Solid Geometry (CSG), and Swept Solid Representation (SSR), in the following texts.

2.1.2 Boundary Representation (B-Rep)

Boundary Representation (B-Rep) is a method to describe shapes using their limits. The model represented using B-Rep is an explicit representation, as the object is represented by a complicated data structure giving information about each of the vertices, edges, and loops and how they are joined together to form the object. The geometry of a vertex is given by its (x, y, z) coordinates. Valero et al. [5] developed a method to yield B-Rep models for indoor planar objects (walls, ceilings and floors). Valero et al. [6] then upgraded their method to detect more objects in an indoor environment. Both Tessellated Surface Representation (TSR) and Polygon/Mesh Representation (PR/MR) can be considered as B-Rep types. A final model of TSR or PR/MR, is represented as a collection of connected surface elements. Oesau et al. [7] leveraged a graph-cut formulation to reconstruct a synthetic building point cloud into a mesh-based model. Representing an object using polygonal facets or mesh is the most popular representation in computer graphics. However, there are some problems with polygon mesh models: 1) Level of detail. High resolution could be unduly complex. An option is to reduce the polygon resolution without degrading the rendered presentation. But by how much? 2) Missing data, i.e. occlusions. Large occluded regions are hardly smoothed. Thus, PR/MP does not guarantee that a group of polygons facets can form a closed mesh model. 3) No sense of volume. It is difficult to extract geometric properties such as the radius of a cylindrical column on a mesh representation.

2.1.3 Constructive Solid Geometry (CSG)

Constructive Solid Geometry (CSG) is a high-level volumetric representation that works both as a shape representation and a record of how an object was built up [8]. The final shape can be represented as the

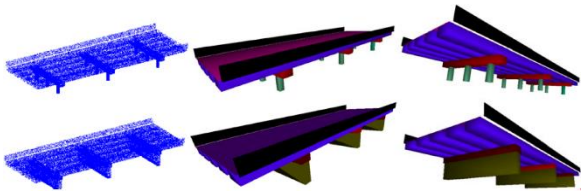


Figure 2. Fitted IFC entities in synthetic bridge point clouds (Zhang et al., 2014)

combination of a set of elementary solid primitives, which follow a certain "logic". The primitives can be cuboids, cylinders, spheres, cones, and so on. When building a model, these primitives are created and positioned, then combined using Boolean set operators such as union, subtract, intersect and so on. Xiao and Furukawa [9] introduced an algorithm called "inverse CSG" to reconstruct large-scale indoor environments with a CSG representation consisting of volumetric primitives. However, this method uses only cuboids as volumetric primitives, assuming that they are the most common shapes found in indoor walls. Zhang et al. [10] (Figure 2) designed a classifier from surface primitive features to classify both infrastructure components (pier, beam, deck etc.) and 3D shape entities labels (cuboid, cylinder, sheet etc.) However, this method is tailored for idealized or simplified topology designs that do not consider the real geometries of bridge components. For example, a real sloped slab with varying vertical elevation cannot be simply modelled by a single sheet.

2.1.4 Swept Solid Representation (SSR)

Swept Solid Representation (SSR) or Extrusion is a representation of model which creates a 3D solid shape by sweeping a 2D profile that is completely enclosed by a contour line along a specific path. The sweeping line could be a straight line perpendicular to the contour surface, or it could be a curve in 3D space. Ochmann et al. [11] presented a method to reconstruct parametric 3D building models from indoor point clouds. Laefer & Truong-Hong [12] introduced a kernel-density-estimated-based method to identify the cross-sections of steel beams in point clouds. A cross-section is matched to one real steel type from a steel standard library and is extruded along the alignment.

2.2 Industry Foundation Classes (IFC)

2.2.1 IFC and MVD

IFC provides a set of definitions for all object element types encountered in the construction sector and a text-based structure for storing those definitions in a data file, based on an open data exchange standard, i.e. the IFC schema. It defines three basic components for modelling constructions: objects, relationships, and properties. An object is an abstract super-type entity, *IfcObject*, structured in an order hierarchy. An instance of the entity is used to represent a real-world object. The concept of relationships is the objectified relationship, *IfcRelationship*, relating different objects to each other, and the property definition, *IfcPropertyDefinition*, is the generalization of all characteristics and context information that can be added to an object. Ji et al. [13] introduced an extension to the IFC-Bridge format, providing a means of interchanging parametric bridge models. They describe in detail the necessary entities introduced to define parameters and capture dimensional and geometric constraints. Likewise, Amann et al. [14] suggested a generalized alignment model that can be extended with cross sections to describe a road body. This model can be further used for other product data models of linear infrastructure contractions, such as tunnels, roads, and bridges.

IFC is huge and defines a detailed schema of roughly 800 data types for representing building objects, their relationships, and associated lifecycle information. Specific uses of IFC have been narrowed to smaller subsets using a fraction of the data definitions, called Model View Definitions (MVD). The SeeBridge research team compiled an Information Delivery Manual (IDM) [16] to ensure that the final bridge DT would be sufficiently semantically meaningful to provide most of the information needed for subsequent bridge repair, retrofit and rebuild work.

2.2.2 IFC Geometric Representation

The section represents the most important IFC geometry representations. According to Borrmann et al. [15], all geometry representations in IFC data model can be grouped into four classes: Bounding Boxes, Curves, Surface models, and Solid models.

Specifically, bounding Boxes can be represented using *IfcBoundingBox*. Bounding Boxes are highly simplified geometric representation for 3D objects that are usually used as placeholders. *IfcBoundingBox* is defined by a placement corner point and dimensions of the three sides as a cuboid. *IfcCurve* and its subclasses *IfcBoundedCurve*, *IfcLine*, and *IfcConic* can be used to model line objects. Freeform curved edges (i.e. splines) and curved surfaces are required to model complex geometries. Surface models are used to represent composite surfaces comprised of sub-surfaces. They can be curved surfaces, such as NURBS surfaces or flat surfaces, such as mesh. *IfcTriangulatedFaceSet* can be used to represent the tessellated surfaces, i.e. polygons with an arbitrary number of edges, or triangular mesh. TSR cannot represent curved surfaces ideally but approximate them into triangular facets. In this case, the curved surface can be described using a finer mesh size, if accuracy is a concern. Specifically, *IfcBSplineSurface* can be used for representing curved surfaces. One classic way to generate 3D objects as solid models is

through CSG. *IfcCsgPrimitive3D* and its subclasses such as *IfcBlock*, *IfcRightCircularCylinder*, *IfcSphere*, and so on can be used. However, the use of CSG is very limited due to the fact that the use of primitives is very restrictive. By contrast, SSR is widely used for creating 3D objects in IFC. Possible representations include but not limited to the classes summarized in the following. In general, IfcSweptAreaSolid and its subclasses *IfcExtrudedAreaSolid*, *IfcRevolvedAreaSolid*, *IfcFixedReferenceSweptAreaSolid*, and IfcSurfaceCurveSwptAreaSolid can be used to present extruded solids. A closed profile *IfcArbitraryClosedProfileDef* is necessary for this representation. When using *IfcExtrudedAreaSolid*, the ExtrudedDirection is defined so that *IfcArbitraryClosedProfileDef* can be extruded along the direction. When using *IfcRevolvedAreaSolid*, both ExtrudedDirection and axis are defined so that *IfcArbitraryClosedProfileDef* can rotate around the axis up to a given angle. Then, *IfcFixedReferenceSweptAreaSolid* allows the extrusion to be done along any curve in space through the attribute Directrix. That is to say, the profile is extruded along a specific axis defined by the attribute FixedReference. By contrast, *IfcSectionedSpine* and *IfcSweptDiskSolid* are two representations working in a different but similar way. Detailed descriptions can be found in [15].

Among existing 3D object fitting work, almost all methods are used for generating building or industrial elements, such as walls, ceilings, floors, and standardized industrial elements. These objects are simply represented as extruded planes elements, cuboids, cylinders, and extruded steel beams. The gDT generation for existing RC bridges is almost missing in the literature. The problem of fitting 3D solid models in IFC format to real bridge point clusters has yet to be addressed. No effective method can reconstruct bridge point clusters into 3D IFC objects. In addition, no standardized metric is available for the quantitative evaluation of a gDT.

3 Proposed Method

3.1 Scope

Our method focuses on four types of bridge components: slab, pier, pier cap, and girder, in typical RC slab and beam-slab bridges. These two types of RC bridges represent 73% existing and 86% planned future bridges in the UK. These four types of components represent the most important and the most detectable structural components in the two types of bridges.

3.2 Overview

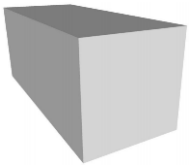
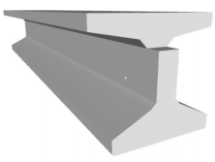
A definition of the level of detail (LOD) for gDT generation of existing infrastructure is missing in the literature. We use the LOD specification suggested by BIMForum as guidance. Table 2 illustrates an example of the interpretation of the LOD for a highway bridge component: a concrete precast girder.

The inputs of the proposed method are the four types of labelled point cluster. The outputs are two IFC files, containing various *IfcObjects* making up a bridge gDT and corresponding to two different LOD: LOD 200 and LOD 250—300. We define the LOD 250 as a LOD that is higher than LOD 200 but may not necessarily totally reach LOD 300. Figure 3 illustrates the workflow of the proposed method, which consists of two major steps: Step 1, geometric feature extraction of point clusters; Step 2, *IfcObjects* fitting to the extracted features. We use the MVD proposed by Sacks et al. [16], which proposes a binding to the IFC4 Add2 standard for exchanging bridge DTs.

3.3 LOD 200 gDT generation

We use TSR to create the Oriented Bounding Box (OBB) for each point cluster. The reason to choose TSR is because the OBB of a point set is a parallelepiped of 12 edges, 8 vertices and 6 faces. TSR is an explicit way to present an OBB. The parallelepiped geometry can be represented using the tessellated geometry model through *IfcShapeRepresentation*, expressing it as a triangulated tessellation. The coordinates of each vertex are provided by an index into an ordered list of Cartesian points *IfcCartesianPointList3D*. We introduce the property set *Pset_BoundingBoxProperties*, in which the method adds the attributes such as the length, width, and height of each OBB and composes them into an *IfcPropertyset*.

Table 2 LOD Specification for Highway Bridge Precast Structural Girder

LOD	Interpretation	Schema
200	Elements are generic placeholders. Any information derived from LOD 200 elements must be considered approximate.	
300	The quantity, size, shape, location, and orientation of the element as designed can be measured directly from the model.	

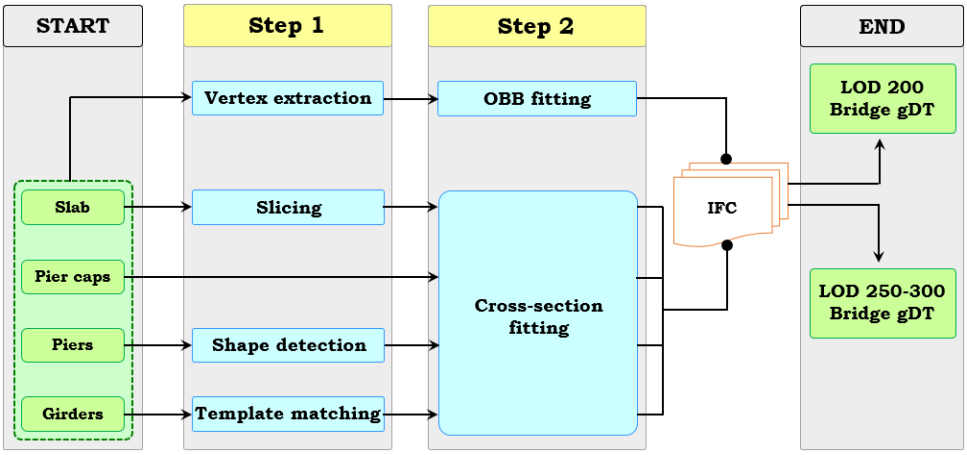


Figure 3. Workflow of the proposed IFC object fitting method

3.4 LOD 250 – 300 gDT generation

Solid extrusions are preferred wherever possible if the cross-section in each slice model is deemed to be constant.

3.4.1 Slab – *IfcSlab*

Real-world bridges are neither straight nor flat. To circumvent or be compatible with the existing constraints of road geometry, many highway bridges carrying roads are on a curved alignment and the supporting structure follows that curved alignment. We use a similar but not identical slicing method to that proposed in [17] to slice the deck slab into J slices. The slicing does not take a parallel pattern but is rather oriented along the normal direction of the slab curved alignment. According to Kobryń [18], a circular curve is assumed to be the horizontal alignment of bridges investigated in this research. We then project the deck slab point cluster onto the XY-plane followed by fitting it with a unique second-degree polynomial to derive the parabola of the deck slab alignment. Next, we compute the tangent, slope, and normal at each interpolant of the parabola. The deck slab is then segmented along the direction of the normal of each interpolated position into J slices (Figure 4).

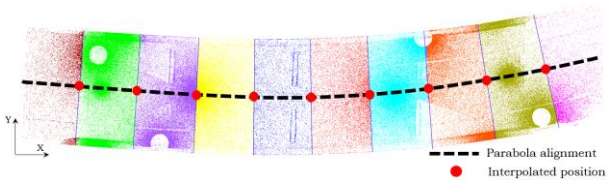


Figure 4. Deck slab slicing

Then, the problem of modelling the whole deck slab is transformed into modelling each straight slab slice assuming each slice is straight along the tangent direction and the cross-section of each slice is constant. For each slice, we rotate the slice around the Z-axis:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(-\varphi_j) & \sin(-\varphi_j) & 0 & 0 \\ -\sin(-\varphi_j) & \cos(-\varphi_j) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad (1)$$

where the rotated angle φ_j is the angle between the normal direction of the alignment of the slice j and the global Y-axis. The updated points (x', y', z') are used to define the cross-section of each slice and descripted using a 2D α-shape. Each hull of the α-shape is represented as a 2D Cartesian point *IfcCartesianPoint*. A profile *IfcArbitraryClosedProfileDef* is used and the slab slice geometry is then represented using an extruded geometry model through *IfcExtrudedAreaSolid* and *IfcShapeRepresentation*, expressing it as a Swept Solid. The ExtrudedDirection is defined using the tangent direction of the mid-point of each slice. We introduce the property set *Pset_SlabSliceProperties*, in which the method can add the attributes of each slab slice.

3.4.2 Pier cap – *IfcBeam*

Similar to how the slab slice is extruded, when modelling a pier cap point cluster, we project its points onto the XY-plane. We then use 2D α-shape to describe the projected contour such that each hull is stored in a 2D Cartesian point *IfcCartesianPoint* followed by mapping the contour with a list of *IfcPolyLine* objects. Like the slab slice, a pier cap is also represented as a Swept Solid through *IfcArbitraryClosedProfileDef* and *IfcExtrudedAreaSolid*.

3.4.3 Pier – *IfcColumn*

Piers can take many configurations. The shape of a pier is defined by the shape of its cross-section. To simplify the problem, we group pier shapes into 3 classes: Shape 1 – Circular, Shape 2 – Quadrilateral, and Shape 3 – Others. Unlike simplified scenarios and synthetic data, real objects embedded in point clouds are similar to hand-drawn geometric shapes that usually contain imperfections or distortions. To tackle this challenge and identify the cross-section shape of a pier, we use a fuzzy-logic-based shape descriptor. We project a pier point cluster points onto the global XY-plane followed by calculating the perimeter of the projected points (denoted P_{ch}) and the bounded area (denoted A_{ch}). We then compute the area of the enclosing rectangle, i.e., the 2D oriented-bounding-box (denoted A_{er}) and the area of their largest-quadrilateral (denoted A_{lq}). if ${P_{ch}}^2/A_{ch} \cong 4\pi$, then the cross-section is a circle; else if $A_{ch}/A_{er} \cong A_{lq}/A_{er} \cong 1$, then the cross-section is a rectangle; Otherwise, the cross-section takes another shape. Similarly, cylindrical pier is represented using *IfcExtrudedAreaSolid* and *IfcShapeRepresentation*, expressing it as a Swept Solid. Attributes such as Position, Direction, Diameter, and Length. Then, stacked representation is used to approximate the

Figure 5. An RC bridge point cloud (L) and the stacked representation (R)

overall pier shape through multiple slice models for quadrilateral and other piers (Figure 5).

3.4.4 Girder – *IfcBeam*

The girders studied in this research are assumed precast as the majority of existing and planned future RC slab and beam-slab bridges in the UK select precast elements. We suggest a template matching method to find the best-match girder type in existing precast bridge beam catalogues. We use three criteria to specify the girder type in each span: 1) Span length *sl*; 2) Girder bottom flange b_f; and 3) Web depth *d*.

We use the span length *sl* to narrow down a possible range of girder types. Then, the averaged girder flange b_f and the web depth *d*, computed using [17] can select a specific girder from the range of girders. Next, we store each of the cross-section feature point of the identified girder in *IfcCartesianPoint*. A 2D profile *IfcArbitraryClosedProfileDef* is used to describe the profile. Similarly, the girder is represented as a Swept Solid.

4 Experiments

4.1 Ground Truth Data and Results

We used the 10 bridge point clouds collected in [17] to conduct our experiments. The raw data is available at: http://doi.org/10.5281/zenodo.1233844. We also manually generated two sets of models *GT A* and *GT B* (Table 3), which serve as ground truth data:

GT A: The four types of bridge components in this set of models were represented using their tightest cuboids (average modelling time: 0.92 hours). They are used to compare against the automated generated LOD 200 gDTs.

GT B: The four types of bridge components in this set of models were represented within their precise dimensions (average modelling time: 27.6 hours). They are used to compare against the automated generated LOD 250-300 gDTs.

We implemented the proposed method on Gygax (https://github.com/ph463/Gygax/) as two different classes according to the suggested two different model resolution on a desktop computer (CPU: Intel Core i7-4790K 4.00GHz, Memory: 32GB, SSD: 500GB).

Table 4 demonstrates the results of the automated gDTs: LOD200 gDTs and LOD250-300 gDTs. Compared to manual modelling times, the average modelling time 10.2 seconds for LOD200 and 37.8 seconds for LOD 250-300, are trivial. We only demonstrate 4 bridge results due to limited space.

4.2 Evaluation

4.2.1 Evaluation of LOD 200 gDTs

We computed the volume and the centroid of each GT cuboids (gtBBox) Vol_{gt} and C_{gtBBox} and the automated ones Vol_{auto} and C_{autoBBox}. Denote E_{dc} and FVR are the Euclidean distance and false volume ratio between each C_{gtBBox} and the corresponding C_{autoBBox}.

$$\text{FVR} = \frac{|Vol_{auto} - Vol_{gt}|}{Vol_{gt}} \quad (2)$$

We also computed the point-to-point (P2P) distance, which is Euclidean distance between each vertex of the automated gDT and that of the GT one of each component of each bridge. The averaged $\overline{\text{P2P}}$ of all the 10 bridges was 23 cm (Table 5).

Table 5 Comparison of LOD 200 gDTs and *GT A*

	E_{dc} (m)	FVR (%)	P2P (m)
Bridge 1	0.06	17.6	0.23
Bridge 4	0.17	10.8	0.19
Bridge 7	0.23	24.1	0.35
Bridge 9	0.18	16.1	0.30

Table 3 Manual modelling *GT A* and *GT B*

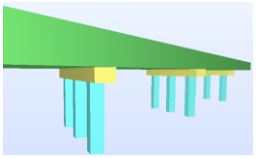
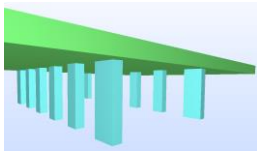
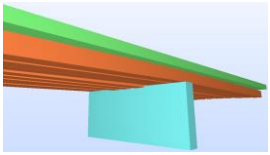
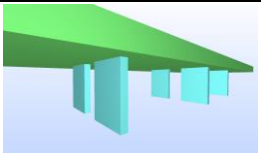
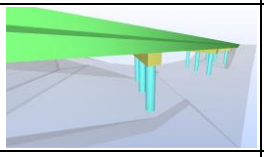
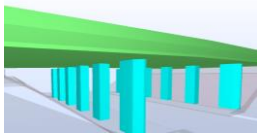
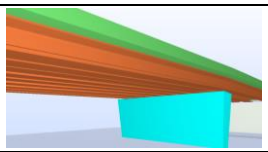
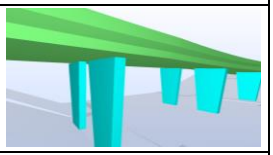
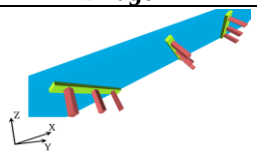
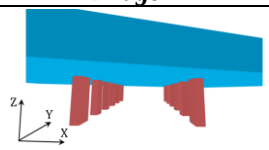
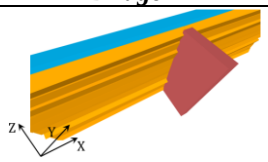
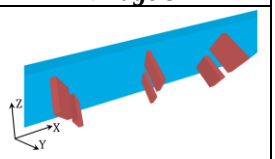
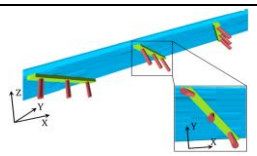
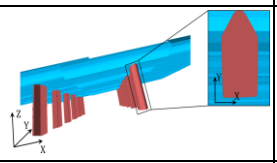
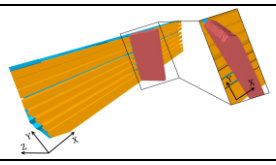
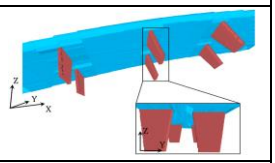
	Bridge 1	***Bridge 4***	***Bridge 7***	***Bridge 9***
GT A				
Time (h)	1.1	1.5	1.75	0.5
GT B				
Time (h)	50.2	25.9	26.6	20.1

Table 4 LOD 200 gDTs & LOD 250 – 300 gDTs

	Bridge 1	***Bridge 4***	***Bridge 7***	***Bridge 9***
LOD200 gDT				
Time (s)	10.1	9.5	8.2	10.0
LOD250 -300 gDT				
Time (s)	25.5	58.1	31.1	37.3

4.2.2 Evaluation of LOD 250 – 300 gDTs

We used a cloud-to-cloud (C2C) distance evaluation to detect changes between *GT B* and the automated ones. To do so, we first converted the *GT B* and the automated gDTs into .obj files followed by random sampling dense points from the generated polygons. Then, both sampled bridge point clouds (denoted GT and Auto) were compared against the reference point cloud, which is each bridge's original point cloud (denoted Real). We followed a local distance strategy to compute a local model Q. A quadratic model is used to fit neighbouring points in the reference point cloud to a surface (radius=0.3m) so that the average local distance from a compared point cloud α to a reference point cloud β is:

$$\overline{\mathrm{dist}} = \frac{1}{n}\sum_{i=1}^{n} \min\{d(q_i, Q)\}, \quad (3)$$

where q_i is a point of the compared point cloud α that is not on the model Q. Then, the estimated C2C distance between the two clouds is the bigger one of the mutual $\overline{\mathrm{dist}}$:

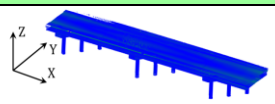
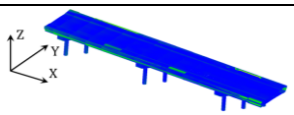
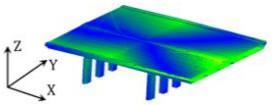
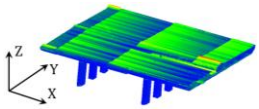
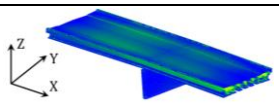
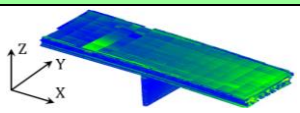
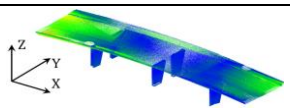
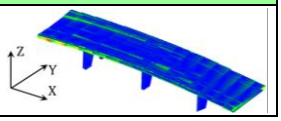
$$\mathrm{C2C} = \max(\overline{\mathrm{dist}}_{\alpha/\beta}, \overline{\mathrm{dist}}_{\beta/\alpha}). \quad (4)$$

An automated gDT is deemed better modelled if its C2C (denoted $\mathrm{C2C_{Auto}}$) is smaller compared with that of the manual model (denoted $\mathrm{C2C_{GT}}$) and vice versa (Table 6).

5 Conclusions

This paper presents an object fitting method able to generate gDTs of existing RC bridges in IFC format, using the four types of point clusters making up the bridge. Compared to the manual modelling process, the proposed method was more consistent, less liable to human errors. The gDTs were evaluated using distance-based quantitative metrics. Most of the automatically generated gDTs were better than the manually generated ones in terms of spatial accuracy. The overall $\overline{\mathrm{C2C}}_{\mathrm{Auto}}$ of 10 bridges gDTs was 7.05 cm while the $\overline{\mathrm{C2C}}_{\mathrm{GT}}$ was 7.69 cm. This value was down to 5.6 cm for $\overline{\mathrm{C2C}}_{\mathrm{Auto}}$ while 7.0 cm for $\overline{\mathrm{C2C}}_{\mathrm{GT}}$, if we didn't take two large distances (*Bridge 7* & *9*) into account. Last, the modelling time was also drastically reduced.

Table 6 Comparison of C2C between *GT E* PCD and Auto PCD against Real PCD

Bridge 1		***Bridge 4***	
$C2C_{GT/Real}$	$C2C_{Auto/Real}$	$C2C_{GT/Real}$	$C2C_{Auto/Real}$
4.0 cm	4.3 cm	7.3 cm	9.4 cm
			
Bridge 7		***Bridge 9***	
$C2C_{GT/Real}$	$C2C_{Auto/Real}$	$C2C_{Real/GT}$	$C2C_{Auto/Real}$
15.7 cm	12.5 cm	9.8 cm	5.6 cm
			

Acknowledgements

This research is funded by EPSRC, EU Infravation SeeBridge project and Trimble Research Fund. We thank their support.

References

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

[2] Network Rail. (2015). Network Rail Bridge List.

[3] Buckley, B., & Logan, K. (2017). The Business Value of BIM for Infrastructure 2017. *Dodge Data & Analytics*, 1–68.

[4] Lu, R., & Brilakis, I. (2017). Recursive Segmentation for as-is Bridge Information Modelling. LC3

[5] Valero, E. et al., (2016). Semantic 3D Reconstruction of Furnished Interiors Using Laser Scanning and RFID Technology. *Jr of Comp in Civil Eng*.https://doi.org/10.1061/(ASCE)CP.1943-5487.0000525

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

[7] Oesau, S., Lafarge, F., & Alliez, P. (2014). Indoor scene reconstruction using feature sensitive primitive extraction and graph-cut. *ISPRS*. https://doi.org/10.1016/j.isprsjprs.2014.02.004\

[8] Deng et al., (2016). Mapping between BIM and 3D GIS in different levels of detail using schema mediation and instance comparison. *Aut in Constr*, https://doi.org/10.1016/j.autcon.2016.03.006

[9] Xiao, J., & Furukawa, Y. (2014). Reconstructing the World's Museums. *International Jr of Computer Vision*, https://doi.org/10.1007/s11263-014-0711-y

[10] Zhang et al., (2014). Automatic Generation of As-Built Geometric Civil Infrastructure Models from Point Cloud Data. In *Comput in Civil and Building Eng*, pp. 406-413

[11] Ochmann et al., (2016). Automatic reconstruction of parametric building models from indoor point clouds. *Computers & Graphics*, https://doi.org/10.1016/j.cag.2015.07.008

[12] Laefer, D. F., & Truong-Hong, L. (2017). Toward automatic generation of 3D steel structures for building information modelling. *Auto in Constr*. https://doi.org/10.1016/j.autcon.2016.11.011

[13] Ji et al., (2013). Exchange of Parametric Bridge Models Using a Neutral Data Format. *Journal of Computing in Civil Engineering*. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000286

[14] Amann et al., (2015). Extension of the upcoming IFC alignment standard with cross sections for road design. *ICCBEI*.

[15] Borrmann et al., (2018). Industry Foundation Classes: A Standardized Data Model for the Vendor-Neutral Exchange of Digital Building Models. https://doi.org/10.1007/978-3-319-92862-3_5

[16] Sacks et al. (2018). SeeBridge as next generation bridge inspection: Overview, Information Delivery Manual and Model View Definition. *Aut in Constr*. https://doi.org/10.1016/j.autcon.2018.02.033

[17] Lu, R., Brilakis, I., & Middleton, C. (2018). Detection of Structural Components in Point Clouds of Existing RC Bridges. CACAIE. https://doi.org/10.1111/mice.12407

[18] Kobryń, A. (2017). *Transition Curves for Highway Geometric Design. Springer Tracts on Transportations and Traffic* (Vol. 14). Cham: Springer International Publishing.