Zhang, Youqi; Xia, Baiqiang; Taylor, Su

High-resolution 3-D geometry updating of digital functional models using point cloud processing and surface cut

Published in:
Computer Aided Civil and Infrastructure Engineering

DOI:
10.1111/mice.13076

Published: 01/01/2024

Document Version
Publisher's PDF, also known as Version of record

Published under the following license:
CC BY

Please cite the original version:
High-resolution 3-D geometry updating of digital functional models using point cloud processing and surface cut

Youqi Zhang | Baiqiang Xia | Su Taylor

Abstract

Point cloud can provide rich 3-D geometric information of structures, and it has been widely investigated for generating digital functional models, such as finite element (FE) model and building information model (BIM). However, for the existing digital models, how to maintain and update the new local geometric changes on the existing digital models has not been sufficiently studied. Therefore, this research presents a method to update the local 3-D geometric changes on digital functional models using point cloud processing and surface cut. The proposed pipeline starts from processing the point cloud of the local structural change for reconstructing the local surface mesh. After that, the surface mesh is aligned to the corresponding location of the existing digital model by point cloud registration. Then, the geometry updating can be achieved by cutting the digital model with the generated surface mesh. To validate the proposed geometry updating method, two case studies were performed on damaged concrete beams with their intact FE models. The results show that the 3-D geometry changes caused by the damage on the edge and vertex can be successfully updated with high resolution on the existing digital models. The results demonstrate the feasibility of integrating the proposed method into structural health monitoring (SHM) systems for further analysis.

1 | INTRODUCTION

Estimating the health conditions of civil structures, such as bridges and buildings, is crucial for their management and maintenance, as well as the sustainability of urban development. To address this issue, structural health monitoring (SHM) systems (Boller et al., 2009) have been developed and deployed on key civil structures over the past decades. Powered by advanced sensing technology (Sony et al., 2019), signal processing methods (Goyal & Pabla, 2016; Zhang et al., 2022), and deep learning (Goodfellow et al., 2016; LeCun et al., 2015), SHM systems can utilize all available data sources acquired from the monitored structures, including 1-D time-series data (e.g., vibration, strain, displacement, temperature), 2-D image data, and 3-D point cloud data, providing comprehensive information on the operational and damage states of monitored structures.

The 1-D time-series data and 2-D image data of structures are the most commonly used in SHM systems. Vibration data, as a typical 1-D time-series data class, are widely used for identifying both global and local structural changes. For instance, Jiang and Adeli (2007) proposed a...
dynamic wavelet neural network for detecting damage of high-rise buildings. Amezquita-Sanchez et al. (2015) developed a wavelet transform-fractality model for detecting and quantifying damage in high-rise buildings. Rafiei and Adeli (2017, 2018) proposed unsupervised deep learning (DL) models for health condition assessments on structures. Abdeljaber et al. (2017) used convolutional neural networks (CNNs) to identify damage of a steel frame. Zhang, Miyamori, et al. (2019) applied CNNs to identify structural changes on bridges. The recent achievements in vibration-based SHM are reviewed in the article (Avci et al., 2021). The 1-D time-series data can also be used for modal identification (Amezquita-Sanchez et al., 2017; Li et al., 2017; Pezeshki et al., 2023), prediction of dynamic responses (Perez-Ramirez et al., 2019), and strain sensing (Oh et al., 2017) for high-rise buildings. For the 2-D image data, Cha et al. (2017, 2018) used Faster-RCNN to identify multiple types of damage on steel bridges and concrete structures. Zhang et al. (2017) and Zhang, Wang, et al. (2019) proposed CrackNet and a recurrent neural network to detect pavement cracks at the pixel level. Li et al. (2019) also applied fully convolutional network (FCN) to detect concrete damage at the pixel level. Pan and Yang (2020) used dual CNNs to detect postdisaster damage and estimate the corresponding repair cost. Zhang (2020) used YOLO-v3 (Redmon & Farhadi, 2018) to detect concrete bridge surface damage. Lydon et al. (2022) proposed an autoencoder to detect bridge damage based on the displacements measured by a synchronized multicamera system. Zhou et al. (2022) used YOLO-v4 to detect multiclass structural damage for damage assessment. Sajedi and Liang (2021) developed an uncertainty-assisted SHM method, which can identify local damage and detection bridge components using Bayesian inference. Liu and Gao (2022) developed a concrete crack detection method based on the laser line model of visual characteristics of images, and the method achieved similar performance compared to the CNN-based methods. Computer vision–based structural damage detection (SDD) methods are reviewed in the article (Dong & Catbas, 2021). By using those 1-D- and 2-D data–based methods, structures can be investigated and damage can be identified and localized successfully.

Besides the 1-D and 2-D data, 3-D point cloud is becoming prevalent in civil engineering due to the ability to provide 3-D information of structures and spatial relationships in different engineering scenes. For instance, Smile and Sarlo (2022) proposed a method to extract structural beam lines from point clouds of steel buildings by 2-D slicing, image processing, and 3-D projection, which was validated in multiple frame structures. Shen et al. (2021) proposed a method to identify 3-D objects in construction sites and estimate the work zone safety. Golparvar-Fard et al. (2011) used point clouds generated from unordered photo collection to monitor changes of 3-D building elements. Gouda et al. (2021) developed a readiness assessment method for highway infrastructure using point clouds acquired from autonomous vehicles. Especially, point cloud is powerful for representing the as-built or as-is states of structures by creating digital functional models, such as finite element (FE) models (Reddy, 2019; Yu & Adeli, 1993) and building information models (BIMs) (Costin et al., 2018; Volk et al., 2014), which provide platforms for analyzing existing civil structures. Methods for directly converting point clouds to digital models have been proposed by many researchers to improve the efficiency of manual work. For instance, Yan et al. (2017) developed a method to automatically convert point clouds to FE models of bridges with all-hexahedral meshes. Xiong et al. (2013) used point clouds to model the structural components (walls, floors, ceilings, windows, and doorways) of indoor environments of buildings. Barazzetti et al. (2015) used point clouds to create BIMs of a castle, which were then meshed as FE models. Castellazzi et al. (2015) created an FE model of a tower of a monumental historical building using a point cloud with voxel meshes, which achieved high consistency of the first bending model to the CAD-based modeling. Hinks et al. (2013) developed a method to convert point clouds to solid FE models via voxelization and then validated the method on a building structure. Li et al. (2022) proposed a registration-free point cloud generation technique using rotating mirrors and validated the method in an indoor environment. Bassier et al. (2019) proposed a semiautomated conversion method from point clouds to FEM meshes with volumetric tetrahedrons and validated the method on a building structure. These methods achieved accurate transformations from point clouds to digital functional models.

However, since SHM is a long-term process that requires tracking structural degradation and updating the digital models to analyze the effects of the detected structural changes to the structures, directly converting point clouds to digital functional models is insufficient and inefficient for understanding the degradation of structural load-bearing capacity. In many long-term monitoring projects, digital functional models already exist and they need geometry updating methods to stay synchronized with the monitored structures for reflecting their performance. Furthermore, structural damage detected using 1-D and 2-D monitoring data lacks 3-D depth information, which limits the quantitative analysis of structural performance using their digital functional models. Therefore, there is a great need for geometry updating methods for digital functional models. To address this gap, Ghahremani et al. (2018) proposed a method for updating the 3D geometry of FE models by using two point clouds of the target structure in both intact and damaged states. The two point clouds need to
represent closed surfaces for generating solid volumes. Differentiation operations are then performed between the two solid volumes to shape the new geometry of the damaged region. This method achieved remarkable results, however, it also has several limitations. First, it requires obtaining two point clouds of the structure in both intact and damaged states, and the intact one is not always available in real-world engineering scenarios. A more practical way is to interact the point cloud of the damaged structure with the existing intact digital functional model for geometry updating, as the example shown in Zhang and Lin (2022), in which a computer vision–based method for 2-D geometry updating of FE models was proposed. Second, obtaining global point clouds of closed surfaces for generating solid volume is challenging for many civil structures, for example, the point clouds of the upper surfaces of concrete bridge girders are not available. In the case of bridge girders for solid volume generation, the point cloud needs to be closed by estimating the points in the upper surface and added to the point cloud. To avoid these additional processing steps, geometry updating methods using local point clouds of open surfaces are needed for fitting more engineering scenarios. Third, using differentiation of solid volumes for damage profiling is computationally costly and it tends to involve undesired regions in the differentiation computation.

To address the above challenges, this paper proposed a new research question: How to effectively update the local geometry change of structures on existing digital models? Subsequently, to answer the proposed question, this research proposed a novel high-resolution 3-D geometry updating method for accurately profiling new local geometric changes in the existing digital functional models, by incorporating point cloud processing and surface cut. The proposed method enables the interaction between the local point cloud of structures and the corresponding existing 3-D digital function models. Compared to the published works of “scan to model” for model generation, the proposed method is designed as a maintenance method for the existing digital models. Point cloud processing algorithms like surface reconstruction and registration are combined with surface cut operation for profiling damage geometry locally on digital models. Different from the existing methods, the proposed method only requires a local point cloud of the damaged structure and the intact (or relatively intact) digital functional model, without the need for a point cloud of the entire structure’s closed surface, which greatly simplifies the workflow. The proposed method realizes geometry updating by local surface cut, which is more efficient than the volume differentiation in the existing method. Overall, the proposed method is designed to be integrated into the SHM systems for tracking structural degradation as the first version of the pipeline for high-resolution geometry updating. Furthermore, it lays the foundation and represents an important step toward the fully automated procedure of digital twinning (Thelen et al., 2023, 2022).

2 | METHODOLOGY

2.1 | General description of the method

The proposed method aims to accurately update the local 3-D geometric changes of structures on their digital solid models’ corresponding structural components by utilizing point cloud processing and surface cut. The proposed method can be used for damaged concrete bridge girders, piers, building beams, columns, stairs, and so forth. An overview of the method is shown in Figure 1. The method requires two input data sources: (1) point cloud that captures the local geometry of structural damage and (2) the digital functional model of the intact structure or substructure. The point clouds of local damage can be obtained either using active scanners or in a passive fashion via photogrammetry. The intact digital functional model of the intact structure or substructure can be created based on the as-built or design drawings. The output of the proposed method is the geometry-updated digital functional model with newly profiled local changes and more accurate local sectional properties.

In this work, the photogrammetry method was selected due to its no requirement of dedicated hardware. After generating the raw point cloud using photogrammetry, a series of preprocessing operations are performed. First, the point cloud is downsampled to an appropriate density to improve computational efficiency and prevent computational resource exhaustion. Next, the noise component of the point cloud is removed via outlier detection. Following this, a scaling operation is performed on the point cloud to make it the same size as the digital model. Subsequently, the mesh surface of the damaged region is reconstructed through triangulation on the point cloud. After that, the local surface mesh is aligned to the digital model using point cloud registration. Finally, the geometry updating is completed by cutting the intact digital model locally with the reconstructed surface mesh.

2.2 | Computer vision algorithms

2.2.1 | 3-D reconstruction via photogrammetry

Photogrammetry methods can use consumer cameras to generate 3-D structures from unstructured images,
eliminating the need for expensive 3-D scanners like laser scanners or LIDAR scanners. This makes photogrammetry an affordable and popular solution for 3-D reconstruction in many real-world scenarios. The typical photogrammetry pipeline involves several steps as shown in Figure 2, including feature extraction, image matching, feature matching, structure from motion (SfM) (Agarwal et al., 2011; Özyeşil et al., 2017; Schonberger & Frahm, 2016), and multiview stereo. First, an image describer extracts features (such as the scale-invariant feature transform [SIFT] describer Lowe, 2004) from each unstructured image. Then, images showing the same areas of the scene are paired using the Vocabulary Tree matching algorithm (Nister & Stewenius, 2006), and the features from each image pair are filtered and matched using geometric and photometric filtering methods. The SfM method is then applied to obtain the point cloud of the scene. Finally, a dense point cloud is generated using multiview stereo, estimating the depth value of each pixel in each image using the Semi-Global Matching method (Hirschmuller, 2007; Hoffmann, 1989). This pipeline can produce dense point clouds of geometric surfaces of structures.

2.2.2 Voxel downsample

Voxel downsampling is a simple yet efficient way to reduce the density of point clouds. The space of a point cloud is split into uniform and continuous voxels, and the points are then embedded in the voxels. Each voxel is represented by a new point calculated by the mean coordinates of the embedded points in the voxel. The only parameter for the simplification is the size of the voxels. The larger the voxel size, the sparser the downsampled point cloud. Figure 3 shows an example of the point cloud of a box before and after downsampling.

2.2.3 Point cloud noise reduction

Raw point cloud collected from devices or generated from photo sets via photogrammetry tend to contain noise components, which may affect the quality of surface reconstruction and registration. Statistical outlier detection (Carrilho et al., 2018) can detect and remove noise points that are far away from their neighbors compared to the average value for the point cloud. Two parameters control the noise filter: number of neighbors and standard deviation ratio. Number of neighbors specifies how many neighbors are considered to compute the mean distance for a given point. Standard deviation ratio is assigned as the threshold based on the standard deviation of the average
differences of the whole point cloud. The lower the standard deviation ratio, the more filtered points.

2.2.4 Point cloud registration

Point cloud registration can be achieved through a two-step procedure: coarse registration and refinement. Coarse registration mainly uses random sample consensus (RANSAC) (Fischler & Bolles, 1981) and fast point feature histograms (FPFH) (Rusu et al., 2009). RANSAC is an algorithm that finds the optimal parameters of a mathematical model by testing a noisy data set. FPFH is a simplified version of a feature descriptor called PFH, which describes the geometric relationship between points.

In the FPFH algorithm, a query point and its $k$ neighbors in a region with a given radius $r$ are pairwise coupled. For each point pair, a local coordinate system $(u, v, w)$ is built based on

$$
\begin{align*}
  u &= n_1 \\
  v &= u \times (p_2 - p_1) \\
  w &= u \times v
\end{align*}
$$

in which $p_1$, $p_2$, $n_1$, and $n_2$ are two points with their normals. Then, three features $\alpha$, $\phi$, and $\theta$ are expressed in terms of the local coordinate system $u$, $v$, $w$, the two points $p_1$, $p_2$, and their normals $n_1$, $n_2$ by the dot product operations as

$$
\begin{align*}
  \alpha &= v \cdot n_2 \\
  \phi &= u \cdot \frac{p_2 - p_1}{\|p_2 - p_1\|_2} \\
  \theta &= \arctan(w \cdot n_2, u \cdot n_2)
\end{align*}
$$

By using those three features, the simple point feature histograms of the query point $SPFH(p_q)$ and its neighbor points $SPFH(p_k)$ can be computed. $SPFH(p_q)$ and $SPFH(p_k)$ count the numbers of the three features $\alpha$, $\phi$, and $\theta$ for the query point and its neighbors in each bin. Then, the final result of $FPFH$ is calculated as

$$
FPFH(p_q) = FPFH(p_q) + \frac{1}{k} \sum_{i=1}^{k} \frac{1}{w_k} \cdot SPFH(p_k)
$$

in which $w_k$ represents the distance between query point $p_q$ and a neighbor point $p_k$, and $k$ is the number of neighbors of the query point $p_q$. The final result considers both the features of the query point $SPFH(p_q)$ and its neighbors $SPFH(p_k)$ to increase the robustness of the algorithm. By the above process, $FPFH$ can describe the local geometric properties of a point by a feature vector, which can be used to search the points with similar local geometric structures in the target point cloud. The FPFH algorithm requests the normals of points, which are estimated by fitting a local plane to each point.

The refinement step of point cloud registration uses a method called Iterative Closest Point (ICP) registration (Rusinkiewicz & Levoy, 2001) to obtain a more accurate transformation matrix $T$. The ICP registration algorithm iterates the following two steps: (1) Extract the corresponding set $K$ from the target point cloud $P$ and the source point cloud $Q$ and (2) update the transformation matrix $T$ by minimizing a loss function $E$ as defined as

$$
E(T) = \sum_{(p,q) \in K} \| p - Tq \|_2^2
$$

The transformation matrix $T \in \mathbb{R}^{4 \times 4}$ includes two main parts: rotation matrix $T_r \in \mathbb{R}^{3 \times 3}$ and translation vector $T_t \in \mathbb{R}^{3 \times 1}$, as formed as

$$
T = \begin{bmatrix} T_r & T_t \\ 0 & 1 \end{bmatrix}
$$

2.2.5 Surface reconstruction

The ball pivoting algorithm (Bernardini et al., 1999; Digne, 2014) is a classic method for surface reconstruction from point cloud. The method considers a 3-D ball with a given radius dropping on a point cloud. When the ball touches any three points, a triangular surface element is generated. Then, the ball pivots from the edges of the existing triangle and iterate the above process, a surface mesh can be created from the point cloud. Figure 4 shows an example of the ball pivoting method. The quality of the surface depends on the radius of the ball. A small radius may lead to holes in the generated surface, while a large radius may create unwanted triangles. The appropriate radius should be selected based on the point density of the cloud.
2.3 Geometry profiling via surface cut

The concept of geometry profiling using surface cut is demonstrated in Figure 5, where a solid volume is cut by a surface. The solid volume can be assumed as an intact structure, while the surface contains the geometric details of the damage. Then, the geometry updating of the damaged structure can be achieved by cutting the intact solid volume with the surface. The intact volume is split into two segments: the geometry updated structure and the volume of damage.

In practical engineering cases, surfaces for cutting are generated from point clouds of structures, and these surfaces are not simplistic planes as in the example shown in Figure 5. A typical reconstructed surface for geometry updating consists of two parts: the region of intact surfaces and the region of damage. The intact region mainly contributes to alignment, while the damaged region is used to profile the geometries of damage on the intact digital models. However, due to the complex geometry of the curved surface reconstructed from point cloud, the surface may not always separate the intact digital model into the desired two parts after alignment and cutting. This is because the reconstructed surface in the intact region is not completely flat, and there are many small fluctuations on it. These small fluctuations may cause many unnecessary small local cuts in the intact region. To prevent this undesired phenomenon, after aligning the reconstructed surface to the intact digital model by point cloud registration, the reconstructed surface is slightly moved out of the intact digital model. This operation can ensure that the small fluctuations of the reconstructed surface in the intact regions do not touch or overlap the solid volume, leading to a neat surface cut.

3 EXPERIMENTS AND RESULTS

To validate the proposed geometry updating method, two case studies are presented in this section. Damaged concrete beams with volume losses are selected as the specimens. The first case involved damage on the edge of the beam affecting two faces, while the second case involved a loss of volume at the vertex of the beam with three faces involved. The reason for selecting these specimens was that damages on edges and vertices are very common in concrete structures, thus the selected specimens can represent common types of damage in civil structures. For example, in the case of concrete bridge girders, damage on the bottom edges can be usually caused by the impact of tall trucks. Similarly, the volume losses on the edges and vertices of aging stair steps can be a common issue. Overall, selecting specimens that represent common types of damage in civil structures is an important step in developing effective strategies for geometry updating.

In the presented two cases, the photogrammetric computation was performed in Meshroom (Griwodz et al., 2021) to generate dense point clouds. Then, point clouds were cropped in MeshLab (Cignoni et al., 2011) manually. After that, point clouds were processed using Python and Open3D API (Zhou et al., 2018) to generate the surface mesh of local damage. Finally, using FE models as examples, cuts of the intact digital model by the local surface mesh were performed in Abaqus to obtain the geometry-updated digital models.

3.1 Case 1: Damage on edge

The specimen is a concrete beam with volume loss on its edge, as shown in Figure 6a. The dimensions of the beam are displayed in Figure 6b. Local close-up photos of the damage are demonstrated in Figure 7, in which the damage region is indicated with a red box and a yellow arrow. The damage is about 50 mm long, 7 mm deep, and 30 mm high. It was caused by collision during transport.
FIGURE 7 Close-up views of the concrete damage in Case 1: (a) view 1, (b) view 2.

At first, in total over 100 photos of the beam were captured from different views, as summarized in Figure 8a. These unstructured photos were used for 3-D reconstruction via the photogrammetry method introduced in Section 2.2.1. Through the feature extraction, image match, feature matching, and SfM, sparse point cloud and camera views were generated as shown in Figure 8b, which also indicate the positions and orientations of cameras. Finally, a 3-D model of the full scene was generated, as shown in Figure 9a. To extract the beam model, cropping the background is needed. After cropping the unnecessary part, the 3-D model of the concrete beam was obtained, as shown in Figure 9b. A close-up view of the damage region on the edge is shown in Figure 9c.

To increase computational efficiency and prevent unnecessary errors (e.g., computational resource exhaustion), only the damaged part of the beam was cropped for further processing. The point cloud of the partial beam with damage only consists of three faces, as shown in Figure 10, because the beam was placed on a white platform and the contact face was not visible. This situation is similar to the concrete girders in actual engineering cases, in that the upper faces of girders cannot be seen owning to the connections to decks or slabs. Meanwhile, there are noisy points in the point cloud in Figure 10. Thus, voxel simplification and noise reduction were performed on the point cloud. The voxel size is 0.0015 for simplification.

For noise reduction, the number of considered neighbors is 50, and the threshold of standard deviation is 1. Those parameters were determined by trial and error. Through this above processing, the number of points in the point cloud was reduced to 26,077 points, which is only 1.6% of the original point cloud model of the beam (1,675,878 points). Point cloud simplification is necessary for smooth geometric updating.

The preprocessing of the point cloud is finalized by scaling the local point cloud to the identical size as the FE model. The scaling operation can be achieved by using either manual or automatic approaches. As the beam is a cuboid, manually measuring and automatically detecting its width and height are both simple. In this paper, we scaled the local point cloud manually with a ratio calculated by dividing the width of the beam in the FE model by the width of the beam in the obtained local point cloud.

By applying the ball pivoting algorithm to the scaled point cloud, the triangular surface mesh of the local
FIGURE 9 Result of 3-D reconstruction in Case 1. (a) Processed dense 3-D cloud of the scene, (b) processed dense cloud of the beam, (c) close-up view of the damage region in the processed dense point cloud.

FIGURE 10 Local point cloud with noise in Case 1.

damage was generated, as shown in Figure 11. The radius of the ball was assigned to 0.95 determined by trial and error. As a result, a surface mesh that contains 26,077 points and 51,520 triangles triangular elements were generated. The surface mesh achieved very high quality, even in the damaged region. Neither hole nor low-quality element with a sharp angle was generated in the surface mesh.

Subsequently, to calculate the transformation matrix $\mathbf{T}$ for aligning the local surface mesh of damage to the intact FE model for cutting operation, the processed local point cloud of the damaged beam was used as a source point cloud (the blue point cloud shown in Figure 12) for registration. As the point cloud registration algorithms request two point clouds (source and target), another point cloud was sampled from the corresponding faces of the local intact FE model uniformly using Python and Open3D, as the target point cloud (the yellow point cloud shown in Figure 12). This target point cloud has an identical number of points to the source point cloud. The procedure of point cloud registration was summarized in Figure 12. Since the source point cloud and the target point cloud use different coordinate systems, these two point clouds do not match the initial state. Then, through coarse registration, the two point clouds were approximately matched. However, a clear gap can still be observed. After that, the ICP refinement was performed, and the gap between the two point cloud was removed, as shown in the last step in Figure 12. The transformation matrix $\mathbf{T} \in \mathbb{R}^{4 \times 4}$ for the refined registration is obtained via the two-step point cloud registration. To better visualize the registration result, close-up views of the ICP registration result are shown in Figure 13. In the damaged region, the blue points are separated from the yellow point apparently, while in the intact region, the blue points and yellow points are blended. To sum up, the point cloud registration obtained a high accuracy as demonstrated in Figure 13.

After obtaining the transformation matrix $\mathbf{T}$, each point in the surface mesh was projected to the corresponding location of the FE model based on

$$\mathbf{S}' = \mathbf{T} \cdot \mathbf{S}$$

in which $\mathbf{S}'$ is the coordinates of the transformed points, and $\mathbf{S}$ is the coordinates of the points in the source point.
To avoid unnecessary small cuts in the intact region, the surface mesh is moved 1 mm out of the FE model in the $x$, $y$, and $z$ directions. Then, the reconstructed surface of damage was well aligned to the intact FE model for geometry updating, as demonstrated in Figure 14. Subsequently, by cutting the instance model using the surface mesh in Abaqus, the geometry of the damage can be successfully updated on the instance model with rich details, as shown in Figure 15. As the digital functional model is an FE model in the case study, the high-resolution geometry-updated FE model was achieved by performing local remeshing, as illustrated in Figure 16, in which the damaged region was modeled with tetrahedron elements, and the other intact region was modeled with hexahedron elements. The element size in the mesh of the damaged region is significantly affected by the point cloud simplification strength. The points in the simplified point cloud were used as nodes in the damaged region of the geometry-updated FE model, and the triangular faces in the reconstructed surface can also form the outside surfaces of the tetrahedron elements in the damaged region.

3.2 Case 2: Damage on vertex

The specimen in Case 2 is a concrete beam with damage on its vertex, as shown in Figure 17. Three faces are involved in the geometric change. The dimensions of the beam are displayed in Figure 17b. Details of the damage on the vertex are shown in Figure 17c. The width, height, and depth of the damage are 121, 25, and 40 mm, respectively.

In total 115 photos of the beam were captured from different views for 3-D reconstruction. Using the photogrammetry approach introduced in Section 2.2.1, feature extraction, image match, feature matching, SfM operations, and MultiView stereo were performed on the photos to obtain the dense point cloud of the scene. After that, A 3-D model of the full scene was generated using MultiView stereo. By cropping the unnecessary part, the background was removed, and only the 3-D model of the concrete beam was extracted. A close-up view of the damaged region on the vertex is shown in Figure 18.

Only the damaged part of the beam in the point cloud was used for further processing, because it can increase computational efficiency and prevent computational resource exhaustion. As the damage is on the vertex, two faces along the longitudinal axis and a face at the end of the beam were involved in the damage region. By performing voxel simplification and noise reduction on the point cloud, the simplified point cloud was obtained, as shown in Figure 19, in which the colors of the points indicate the order of the points in the point cloud. Through simplification, the number of points in the point cloud was reduced from 1,740,636 to 34,125 points, which is only 1.96% of the original cloud of the beam. Then, the simplified local cloud was scaled to have an identical size to the FE model. Using the identical approach as Case 1, we scaled the local cloud manually with a ratio calculated by dividing the width of the beam in the FE model by the width of the beam in the obtained local point cloud.

Subsequently, the triangular surface mesh (shown in Figure 20) was generated using the ball pivoting algorithms as introduced in Section 2.2.5. The radius of the ball was set as 2, and a surface mesh with 51,520 triangles were generated. Same to the surface mesh in Case 1, very high quality of the surface mesh.
Figure 14: Procedure of the damage surface alignment and cutting in Case 1.

Figure 15: Geometry updated instance model in Case 1.

Figure 16: Geometry updated finite element (FE) model in Case 1.

Figure 17: General information of the concrete beam in Case 2: (a) photo of the beam, (b) dimensions of the beam, (c) close-up views of the concrete loss in Case 2.

Figure 18: Close-up view of the damaged region in the processed dense point cloud in Case 2.

was obtained, even in the damage region. No hole was generated in the mesh, and no low-quality element with very sharp angles was presented.

Meanwhile, to align the local surface mesh of damage to the intact instance model for cutting operation, the transformation matrix $T$ was calculated using point
Cloud registration. The input for the registration algorithms includes (1) the simplified local point cloud of damage as the source point cloud and (2) another point cloud sampled from the intact FE model as the target point cloud. This target point cloud and the source point cloud have an identical number of points. The procedure of point cloud registration in Case 2 was summarized in Figure 21, in which the target point cloud is marked yellow and the source point cloud is colored blue. As the source point cloud and the target point cloud use different coordinate systems, these two point clouds did not match in the initial state. Then, after the coarse registration, the two points cloud were approximately matched. However, a clear gap can still be observed between the two points cloud, especially at the end faces of the two point clouds. After that, the ICP refinement was performed, and the gap between the two point cloud was removed, as shown in the last step in Figure 21, indicating an accurate transformation matrix $T$.

After the transformation matrix $T$ was obtained, each point in the surface mesh was projected to the coordinate of the FE model using Equation (6). To prevent unnecessary cuts in the intact region, the surface mesh was moved 1 mm out of the solid FE model in the $x$, $y$, and $z$ directions for a safe distance. Then, the reconstructed surface of damage can be well aligned to the intact FE model for cutting, as shown in Figure 22. By cutting the intact instance model with the surface mesh of the local damage in Abaqus, the geometry of the damage was successfully updated on the instance model with rich geometric details, as shown in Figure 23. Since the digital model is an FE model in this case study, local remeshing was performed as post-processing to achieve a high-resolution geometry updated FE model, as illustrated in Figure 24, in which the damage region was modeled with tetrahedron elements, and the other intact area was modeled with hexahedron elements.

4 | DISCUSSIONS

4.1 | Damage volume estimation

To assess the performance of the proposed geometry updating method, one index can be the volume of damage in the updated digital model. Thus, we compared the volumes of damage in the concrete beams and the corresponding digital models in the presented two case studies. The volumes on the concrete beams were estimated by using the approach illustrated in Figure 25. At first, a liter of fine aggregate (0.1–0.6 mm) was compacted and weighed, and its weight was 1717.92 g. Then, concrete form boards and brackets were installed at the damaged region of the beams. After that, the fine aggregate in the measuring cylinder was used to fill the damage. Finally, the volume of the damage can be estimated based on the reduction of aggregate weight for filling damage as defined in

$$v = \frac{m_0 - m_1}{m_0}$$  (7)

in which $v$ is the volume of damage, $m_0$ means the weight of 1 L of aggregate, and $m_1$ indicates the weight of the remaining aggregate after filling the damage. The damage volume of the updated FE model can be calculated by the difference between the volumes of the instance or FE models before and after geometry updating in Abaqus.

The results of volume estimation of the two case studies are summarized in Table 1. The weights of the filled aggregate are 4.09 g and 90.28 g, respectively. The size of the damage in Case 1 is much smaller than that in Case 2. The ratio of volume is calculated by dividing the simulated damage volume by the measured damage volume.
of the concrete beam. Based on the results in Table 1, the simulated damage volumes are smaller than the measured damage volumes in the actual concrete beams. Case 1 obtained a 67.52% volume ratio, and Case 2 achieved a 94.1% volume ratio. The errors are mainly because of the inaccurate depth information of the generated point clouds in the damaged regions by photogrammetry, rather than the proposed geometry updating process. The study
selected the photogrammetry approach for point cloud acquisition of the local damage because of its simplicity. As the complex geometry in the damaged region, the captured depth information may not be very accurate when the damage is relatively shallow. Comparing the results in Cases 1 and 2, when the depths of damage increase from 7 mm to 25 mm, the volume ratios increase dramatically. When measuring small damage, using a more advanced point cloud generation approach (such as laser scanning) can improve the accuracy by more accurate depth information in the generated point cloud.

Another minor reason for the error is that, in order to ensure that an enclosed damage volume can be obtained when cutting the solid volume intact model with the surface mesh, the surface mesh was moved 1 mm out of the solid FE model in the \(x\), \(y\), and \(z\) directions. Such an operation also compromises a small portion of the damage volume. However, the effect becomes very tiny with the increase in damage scale. Overall, the obtained volume ratios in the two case studies are satisfactory in the current phase of the research.

### 4.2 Effect of point cloud simplification strength on geometry updating precision

Geometry simplification is an important step for improving the computational efficiency when using digital models with complex morphology (Jin et al., 2021). However, it is unknown whether the strength of point cloud simplification of the beam has an obvious effect on the precision of geometry updating of the damaged region. To answer this question, the local point cloud of the beam in Case 2 was simplified with six different voxel sizes: 0.002, 0.003, ..., 0.007, and performed the geometry updating procedure introduced in Case 2 for six times, respectively. The geometry updating results were then compared and discussed in Table 2, which summarizes the details of point cloud simplification and obtained volume ratios based on the six-point cloud simplification strengths. The volume ratio is the ratio between the damage volumes in the updated solid instance model and the actual concrete beam. Originally, the local point cloud of the damage part of the beam consists of 133,636 points, through the six levels of voxel simplification, the numbers of points were reduced in a range between 3283 and 34,125, and the ratios of reduced points were in a range between 74.46% and 97.54%. When the voxel size is 0.002 with the smallest strength of simplification, the damage volume in the updated digital model achieved the highest volume ratio 94.08%. It can be explained because it contained the most geometric details. However, with the increase in the strength of point cloud simplification, the ratio of volume does not show obvious decrease. It fluctuates between 92.67% and 93.15%. The results of geometry updating using the surface meshes generated by the point cloud simplification strengths of 0.002 and 0.007 are visualised in Figure 26 as examples. The geometric details of damage decrease from Figure 26a to Figure 26b. The sizes of triangles and polygons in the damaged surfaces increase gradually through the six scenarios. However, the overall geometry of the damage remains stable, leading to a relatively stable damage volume in the updated FE model. When the point cloud simplification strength is within a certain range, which means that the strength of simplification is not extremely high, the overall morphology of the damage can be maintained, and the volume of the damage remains relatively stable. As the geometry updated instance model will be followed by some postprocessing such as remeshing, the selection of voxel size for point cloud simplification should be based on the environment of the content, like the mesh or element size in the intact FE model.

In general, high point cloud density requests high computational cost, and low point cloud density may compromise the geometry accuracy. How to select a suitable
TABLE 2  Effect of point cloud simplification on damage volume in geometry updated finite element (FE) model.

<table>
<thead>
<tr>
<th>Voxel size</th>
<th>Number of points</th>
<th>Ratio of point simplification (%)</th>
<th>Simulated damage volume (mm(^3))</th>
<th>Measured damage volume (mm(^3))</th>
<th>Ratio of volume (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.002</td>
<td>34125</td>
<td>74.46</td>
<td>49442</td>
<td>52551.9</td>
<td>94.08</td>
</tr>
<tr>
<td>0.003</td>
<td>16351</td>
<td>87.76</td>
<td>48907</td>
<td>52551.9</td>
<td>93.06</td>
</tr>
<tr>
<td>0.004</td>
<td>9508</td>
<td>92.89</td>
<td>48701</td>
<td>52551.9</td>
<td>92.67</td>
</tr>
<tr>
<td>0.005</td>
<td>6190</td>
<td>95.37</td>
<td>48922</td>
<td>52551.9</td>
<td>93.09</td>
</tr>
<tr>
<td>0.006</td>
<td>4399</td>
<td>96.71</td>
<td>48936</td>
<td>52551.9</td>
<td>93.12</td>
</tr>
<tr>
<td>0.007</td>
<td>3283</td>
<td>97.54</td>
<td>48954</td>
<td>52551.9</td>
<td>93.15</td>
</tr>
</tbody>
</table>

Note: The number of points before simplification is 133,636.

density of point cloud is similar to the question: “How many elements are needed in a FE model to model the damage?” The accuracy of the damage geometry and the number of elements in an FE model can be subjective and context-dependent. The suitable density or simplification strength of point cloud depends on how the updated digital models will be used, what are the downstream tasks, the requirements for accuracy and efficiency, and so forth. If time and computational resources allow, higher geometry accuracy can be pursued by using a low point cloud simplification strength and finer local damage geometry. Otherwise, a higher point cloud simplification strength and coarser local damage geometry can be chosen for higher computational efficiency. More specifically, when using the updated model for global structural behavior analysis, such as natural frequencies and mode shapes, high point cloud simplification and coarse local geometry can be used. In contrast, when using the updated model for local structural behavior analysis, such as stress, strain, contact, and corrosion, it is recommended to use a low point cloud simplification strength for finer local damage geometry.

In SHM systems and digital twin systems, digital models can be built in a hierarchical manner with different strengths of point cloud simplifications. Then, based on the downstream tasks, a digital model with suitable point cloud density can be selected.

4.3 Manual steps and automation solutions

The ultimate goal is to fully automate the proposed workflow, so that the method can be integrated into digital twin systems. As this research is in the initial phase, in this paper, the workflow of the geometry updating method includes several manual steps, such as cropping the background, scaling the point cloud, determining of point cloud simplification strength, setting the ball radius, and setting the threshold for noise reduction. The complete automation of the proposed method requires a significant amount of engineering work, thus, the automation of those steps can be subtopics of this project, and the possible solutions are listed as follows.

Cropping operation can be automated by training a DL model for detecting the damaged region, desired structural components, and background. Then, the damaged region and the desired structural components can be extracted from the background. After that, a local coordinate system can be established on the detected structural component, and the coordinates of the interested damaged region for cropping can be determined. Using the relative location of the damage on the structural component, the target point cloud of the intact digital model can also be extracted from the intact digital model.

Selection of ball radius can be automatic by estimating the correlation between the density of points and the suitable ball radius. Then, a surface mesh quality evaluation method is used to detect whether holes and low-quality elements are on the surface mesh, so that whether the selected ball radius is suitable can be evaluated.

Scaling of the point cloud can be automated by training a DL model for detecting the key points in the structure, such as the vertices of beams. Then, the dimensions can be identified and the scale factor can be calculated automatically.

Determination of the point cloud simplification factor can be automated by estimating the desired element size and element number in the generated surface mesh, and then the number of points in the simplified points can be roughly calculated.

Noise reduction can also be automated by training a DL model to identify the noise points, which will be removed after being identified.

5 CONCLUSIONS

This paper presents a high-resolution 3-D geometry updating method for synchronizing the new damages on the
existing digital functional models (e.g., FE model and BIM). Volume losses on structures caused by damage can be captured by 3-D point cloud and converted into a surface mesh, which then profiles the detailed damage geometry on the digital model by surface cut. The proposed geometry updating method is developed to be integrated into the SHM systems, which can be used after damage detection. The proposed geometry updating method forms the pipeline for the synchronization of the local geometry changes of structures on the corresponding digital functional models.

Through the two case studies, several insights can be drawn as follows. (1) The proposed method has been demonstrated to be effective and feasible in updating FE models with rich local geometry details, which can be used for further statistics and estimation of the structural health state. (2) The method only required the point cloud of the local open surface in the damaged region, thus the computational efficiency outperforms the existing methods, which require point clouds of closed surfaces or solid volume of damage. (3) Damage geometry profiling can be accomplished accurately by cutting the intact model with the surface mesh of damage, avoiding complex Boolean operations between solid volumes.

As this research is in the initial stage, the workflow of the method includes manual interference, future works will focus on fully automating the proposed method as discussed in Section 4.3, which can contribute to the computational engine of digital twinning for geometry updating. Meanwhile, as this series of work mainly focuses on the geometry updating of digital models, how to make a strategy to balance the precision of geometry updating, mechanical property calibration, and computational efficiency will be investigated.

ACKNOWLEDGMENTS
The project is financially supported by the Academy of Finland (decision number: 339493). We thank the Aalto studios for the support of equipment. The authors also greatly appreciate the kind support and valuable comments from Mr. Rui Hao, Prof. Jarkko Niiranen, Dr. Tuukka Takala, Mr. Pertti Alho, Mr. Jukka Piironen, Dr. Ruijing Yang, and Dr. Esko Sistonen.

CONFLICT OF INTEREST STATEMENT
The authors declare no potential conflict of interest.

REFERENCES


How to cite this article: Zhang, Y., Xia, B., & Taylor, S. (2024). High-resolution 3-D geometry updating of digital functional models using point cloud processing and surface cut. Computer-Aided Civil and Infrastructure Engineering, 39, 3–19. https://doi.org/10.1111/mice.13076