Inspecting structural components of a construction project using laser scanning

Truong, L.; Lindenbergh, Roderik

Publication date
2020

Document Version
Final published version

Published in
EG-ICE 2020 Workshop on Intelligent Computing in Engineering, Proceedings

Inspecting structural components of a construction project using laser scanning

Linh Truong-Hong[1*], Roderik Lindenbergh[1]
[1]Department of Geoscience and Remote Sensing,
Delft University of Technology, The Netherlands
[*] l.truong@tudelft.nl

Abstract. In construction projects, inspection of structural components mostly relies on typical measurements (e.g. measuring tapes, levelling or total stations). Additionally, with those methods, only a few points on the structure can be measured, and resulted inspection may not fully reflect the actual, detailed condition. Laser scanning is emerging a remote sensing technology to capture the structures' surfaces in high details accurately and quickly. However, because of complex, massive data points acquired from the construction project, in practice, data processing is still manual work with support of computer aided program. To improve current workflows, this paper proposes a method automatically extracting structural components of the concrete building and subsequently inspects them in a term of deformation. The proposed method explores both spatial information of a point cloud and contextual knowledge of structures (e.g. orientation or shape). Additionally, based on the fundamental design of the structures, component's boundaries are automatically extracted to establish un-deformed surfaces of the components for deformation measurement.

1. Introduction

In construction projects, defects of structural components are inevitable. The construction cost can be up 16% of the total cost of the project when the defects were fixed at the last stage (Burati and Farrington, 1987). The rework costs can be minimized if any defect of the component can be identified at an early phase of the project. In current practice, defect inspection is mostly manual interpretation of geometric data acquired from measuring tapes, levelling or total stations. Therefore, project managers cannot identify the defects timely, accurately and objectively. In addition, inspection results in hard copies are cumbersome to use digital tools to improve efficiency of the project management.

A terrestrial laser scanning has ability to capture visible surfaces accurately, quickly and efficiently, which has been used at many construction projects in recent years for construction progress monitoring, surface defect detection, as-built BIM, and dimensional quality control (Bosche, 2010; Kim et al., 2014). As the point cloud represents multiple objects in a scene, a point cloud processing is often required to extract the point cloud of each object and/or objects' surfaces, but this is a nontrivial task. In practice, users often use commercial software (e.g. Revit, and ClearEdge 3D) to manually extract a point cloud of the components by using software functions like crop, segmentation and fitting. That is time consuming and is required experience users to handle a massive, complex data set of the construction project.

Recent effort in automatically extracting the structural components from the point cloud, Bosche (2010) mapped a 3D CAD model to the point cloud to extract members of an as-built model. Dimitrov and Golparvar-Fard (2015) proposed a region growing based on surface roughness estimated for each point by using multi-scale neighborhood to extract building objects. The method still had some drawbacks, for example over-segmentation, intensive computation demand. Kim et al. (2014) extracted data points of surfaces, and edges and corners to determine the dimensions of a reinforced concrete panel, and then as-design model was mapped to the point cloud to assess a construction deviation. Additionally, in supporting quality assurance and control of reinforced structures, Wang et al. (2017) extracted individual rebars

in concrete structures by using one-class support vector machine approach based linearity, planarity and red-green-blue colors of the point cloud. Laefer and Truong-Hong (2017) used kernel density estimation (KDE) to detect primary surfaces of a steel cross-section to determine its shape and dimensions, and subsequently a correct section was identified by comparing the standard sections to the point cloud-based section for generating a 3D model of a steel member. In extracting structural members of a regular building, Maalek et al. (2019) extracted roughly horizontal planes (floor and ceiling) using a histogram based on points' elevation and then a hierarchical clustering method was segmented based on planar and linear features of each points computed from a robust principal analysis. Next, the planar surfaces with normal vectors perpendicular to main axes of the building were considered as columns' surfaces if they make to adjacent surface a symmetric section. In summary, existing methods are requirement of an as-design model, certain assumption to extract the structural components and time consuming. This paper proposes a new method to automatically extract point clouds describing the building components and inspect construction quality in a term of deformation.

2. Proposed method

The proposed method consists of two parts: extraction and inspection of structural components (Fig. 1), in which the first part includes 3 consecutive moduli: (i) floor/ceiling and wall, (ii) column and (iii) beam. The algorithm is based on a synergy between a point cloud, and contextual knowledge of structures. In a construction project of reinforced concrete buildings, structural elements of the story must be inspected before starting a next one. That implies a point cloud of one story is input data for the proposed method rather than those of an entire building. Moreover, contextual knowledge of the structures used in this method includes: (1) floor/ceiling and wall is a planar surface; (2) columns' orientations are vertical; (3) a beam is connected between two columns or a column and wall; (4) the minimum cross-section of the column and beam is 0.2mx0.2m while the columns' height and the beams' length is 2.0m and 1.0m, respectively.

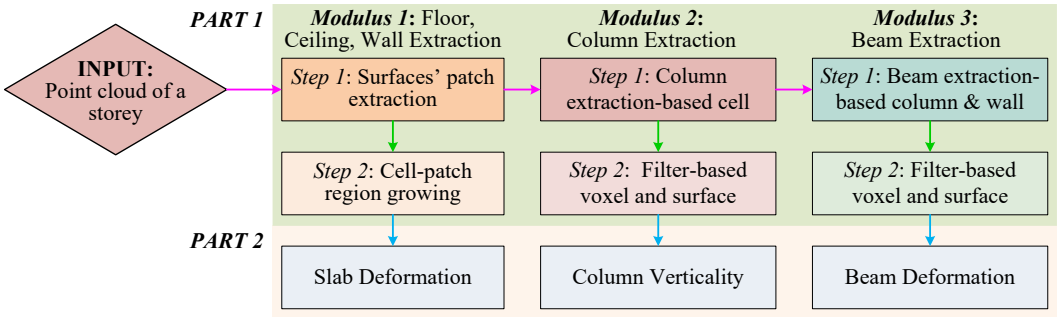


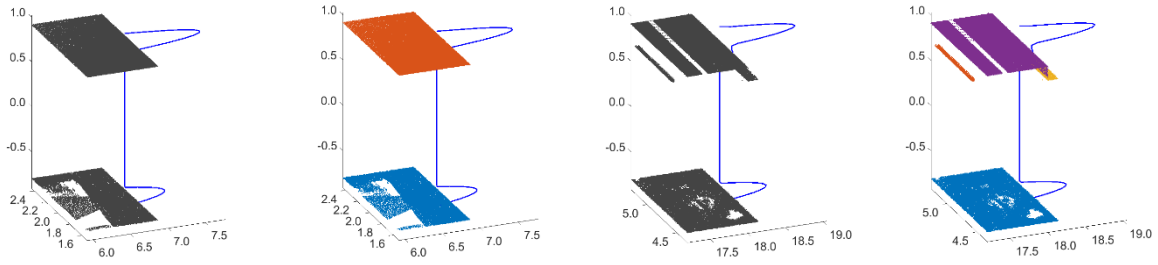
Figure 1: Workflow for inspecting structural components in reinforced concrete buildings

3. Structural component extraction

3.1 Modulus 1: Floor/ceiling and wall extraction

Cell-based segmentation method proposes to extract floor, ceiling and wall as they are planar surfaces in either horizontal or vertical direction. The method consists of two steps: extract the point cloud surfaces' patches through two-dimensional (2D) cells (*Step 1*), and segment a point cloud of a planar surface (*Step 2*).

Step 1: The algorithm employs a quadtree to recursively subdivide an initial 2D bounding box of a point cloud ($P = p_i \in R^3$) into 2D cells ($C = \{c_1, \ldots c_i, c_N\}$, $i = [1, N]$) along the x- and y-directions in a Cartesian coordinate system until the cell size is no larger than the predefined cell size (*cell_size*). The cell c_i is classified as the "*full*" cell if it occupied the number of points larger than a predefined minimum number of the points (*min_ptc*), otherwise, it is an "*empty" cell.* Notably, only full cells are used in a further process. As the cell c_i may contain a point cloud of multiple components (e.g. floor, ceiling, beam or column) in elevation, kernel density estimation (KDE) generated from the z-coordinates of the points is used to extract the data points affiliated to a surface's patch ψ_{ij} (Laefer and Truong-Hong, 2017). The points belonging to the patch are located within two consecutive valleys of KDE (Fig. 2).



a) Case 1: A cell contains data points of a floor and ceiling

b) Case 2: A cell contains data points of a floor, ceiling and beam

Figure 2: Extracting data points of patches within a 2D cell

Step 2: Cell-patch region growing (CpRG) is developed to segments patches representing slabs of the floor and ceiling. CpRG consists of 3 sub-steps: patch-patch region growing (*Step 2.1*), patch filtering (*Step 2.2*), and patch-point region growing (*Step 2.3*).

Step 2.1: Patch-patch region growing

First, the proposed method computes salient features of each patch ψ_{ij}, which include a fitting plane $\psi_{ij}(p_{ij,0}, n_{ij})$, where $p_{ij,0}$ is a centroid of the points $p_{ij} \in \psi_{ij}$ and n_{ij} is a normal vector, and a residual value (r_{ij}) defined as a root mean square of distances $d(p_{ij}, \psi_{ij}(p_{ij,0}, n_{ij}))$. Next, patch-patch region growing starts to segment the patches with an initial seeding patch $\psi_{ij} \in c_i$ owning the smallest residual value to search neighbouring patches ($\psi_{kl} \in c_k$). The patch ψ_{kj} is added to a region (R_k) if it satisfies Eq. 1 and is added to the seeding patch set for next iterations if its residual value r_{kl} is smaller than the residual threshold (r_0). The growing process is completed when all predefined patches are checked. For details of the patch-patch region growing can refer to (Rabbani et al., 2006, Truong-Hong et al., 2018).

$$\begin{cases} \angle n_{ij}, n_{kl} \leq \alpha_0 \\ d\left(p_{kl,0}, \psi_{ij}(p_{ij,0}, n_{ij})\right) \leq d_0 \end{cases} \quad (1)$$

where $d\left(p_{kl,0}, \psi_{ij}(p_{ij,0}, n_{ij})\right)$ is a Euclidean distance between the centroid of the patch ψ_{kj} to $\psi_{ij}(p_{ij,0}, n_{ij})$, and α_0 and d_0 are the angle and distance thresholds.

However, the patches can contain the data points of other components, and/or points of the surface are possessed by unsegmented patches or an adjacent region. These issues can be solved in *Step 2.2* and *Step 2.3*, respectively.

Step 2.2: *Patch filtering algorithm*

For each region, the filtering algorithm extracts a boundary patch ($\psi_{ext,ij}$) of a region R_m and its neighbour interior patches ($\psi_{int,kl}$). Data points of the patches $\psi_{int,ij}$ are used to estimate a local surface (S_{ij}) by using a principal component analysis (PCA). Subsequently, the points $p_{ij} \in \psi_{ext,ij}$ are considered as inlier points $p_{int,ij}$ if the distance $d\left(p_{ij}, S_{ij}(p_{ij,0}, n_{ij})\right)$ is no larger than the distance threshold d_0, and remaining points are eliminated.

Step 2.3: *Patch-point region growing*

The process starts with a boundary patch $\psi_{ext,ij} \in c_i$ of a region R_m to search adjacent patches $\psi_{kl} \notin R_m$. Next, points p_{kl} are added to the region R_m if the distance $d(p_{kl}, \psi_{ij,ext}(p_{ij,0}, n_{ij}))$ is less than the distance threshold d_0. The patch ψ_{kl} is considered as a boundary patch for next searching iterations if more than 50% of the points of the patch ψ_{kl} is added to the region R_k.

Once the data points of the floor and ceiling are extracted, the remaining points cloud are used to extract vertical walls by using a similar process, in which the quadtree is used to generate 2D cells in the *yz* and *xz* planes, respectively. Moreover, details of cell-based segmentation can refer to Truong-Hong and Lindenbergh (2020).

3.2 Modulus 2 - Column Extraction

After extracting floor, ceiling and wall, 2D cells in *xy* plane used for extracting the floor and ceiling are recalled but the segmented points are discarded out of the cells. The algorithm has two main steps (Fig. 3).

In *Step 1*, for each cell c_i, the cell height (H_{ci}), which is computed from *z*-coordinates between the lowest and highest points, and the maximum gap (ΔH_{ci}), which is different *z* coordinates of two consecutive points in an elevation are computed. Next, the cells can possess points of the column if $H_{ci} \geq H_0$ and $\Delta H_{ci} \leq 0.5H_0$, where H_0 is the minimum column height. Subsequently, the remaining cells are clustered based on their connectivity and each cluster contains candidate points of columns (Fig. 3 a and b).

In *Step 2*, a voxel-based growing segmentation (VRG) (Vo et al., 2015) is adopted to segment points of a cluster. In this method, the point cloud is subdivided into the small voxel with the voxel size is no larger than the threshold (*voxel_size*), and then a fitting plane is estimate through points of the voxel to determine voxel's features: a normal vector and residual. VRG operates similar CpRG, in which the voxels associated with salient features used instead of the patches (Fig.3c).

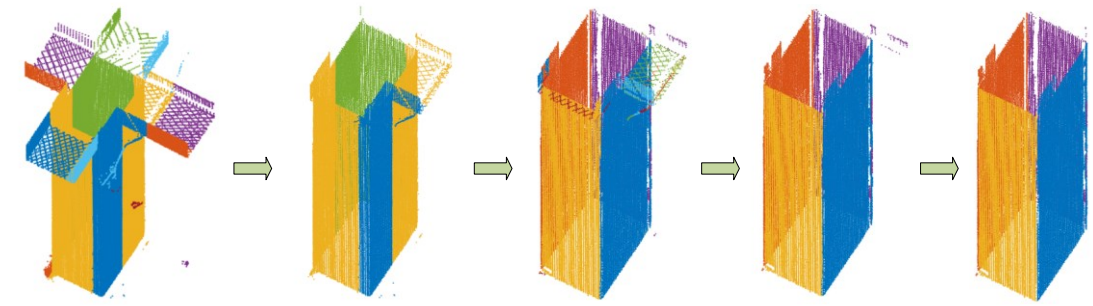
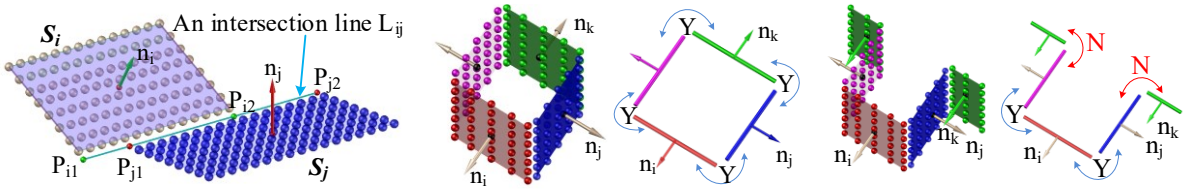


Figure 3: Illustration of Column extraction workflow

As the cluster in *Step 1* may contain points of other components adjoined to the column or may not represent to a real column, a connected surface component (CSC) algorithm is proposed to determine segmented surfaces are parts of the column (Fig. 3d). This is based on the hypothesis is that the component' surfaces connect together in a form of a close loop. The CSC algorithm starts with the initial segment, which is herein the largest surface in a term of an area. Two surfaces are connected if the angle between two normal vectors is larger than the angle threshold α_1, and the overlap length is no less than H_0 (Fig. 4a). Notably, lines $P_{i1}P_{i2}$ and $P_{j1}P_{j2}$ are respectively line segments generated from projected points of the surfaces S_i and S_j on the intersection line L_{ij}.



a) Surfaces connected b) Case 1: all planes accepted c) Case 2: 2 planes rejected

Figure 4: Illustrate the CSC algorithm

Once the connected segments are obtained, their segments are projected on a plane perpendicular to the surface of the reference segment. Subsequently, the segments that do not make a close loop are rejected (Fig. 4b and c). Finally, the column is considered as the real column if they are in a form of at least two surfaces. This criterion is applied to avoid missing the column at the building corner.

Finally, surface-based filtering (SbF) proposes to remove outlier points of the segments, which are points of co-planar surfaces of adjacent components including in the segment. This is based on an observation that the surface's points are bounded by intersection lines with the adjoined surfaces. For the surface S_i, the intersection line L_{ij} divides the points $p_i \in S_i$ into inlier ($p_{inl,i}$) and outlier ($p_{ext,i}$) groups, which is determined based on signs of sign distances $d(p_i, L_{ij})$. The outlier group has small number of the points ($|p_{ext,i}| < |p_{inl,i}|$), and the outlier points $p_{ext,i}$ are then discarded (Fig. 3e).

3.3 Modulus 3 - Beam Extraction

In a building structure, a primary beam connects between two columns or a column and a shear wall, and a beam connects to a side of a column. Moreover, the columns' shapes are mostly rectangular or square, which implies a beam's width is smaller or equal to the column's width. Notably, in this study, the secondary beam connecting two primary beams, is out of a scope of work. Similar the column extraction, the algorithm also consists of two Steps: rough and fine extraction of the beam.

Step 1: From each surface S_{ij} of a column Col_i, the fitting plane $S_{ij}(p_{ij,0}, n_{ij})$ is estimated using PCA, in which a direction of the normal vector is set outward the column (Fig. 5). A beam can connect between the column Col_i and Col_k, if the surface $S_{kl}(p_{kl,0}, n_{kl})$ of the column Col_k satisfies Eq.2. The first constrain is to ensure S_{ij} and S_{kl} parallel, the second constrain checks if Col_i and Col_k are on the same grid, and the last one finds S_{kl} is the closest surface of S_{ij}.

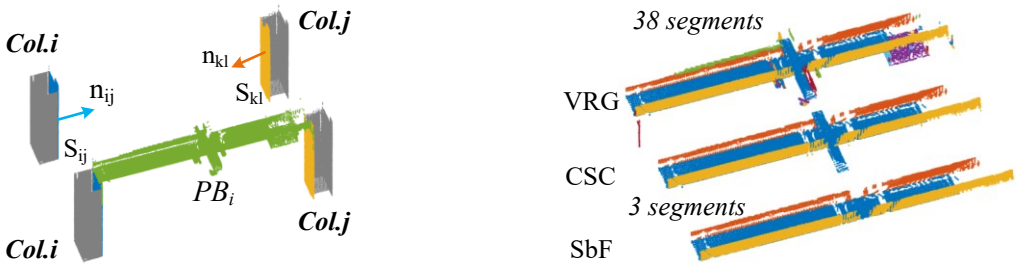
$$\begin{cases} \angle n_{ij}, n_{kl} \leq \alpha_1 \\ \left| d\left(p'_{kl,0}, L_{ij}\left(p'_{ij,0}, n'_{ij}\right)\right)\right| \leq S_{ij}.w \\ d\left(p_{kl,0}, S_{ij}(p_{ij,0}, n_{ij})\right) < 0 \ \wedge \left| d\left(p_{kl,0}, S_{ij}(p_{ij,0}, n_{ij})\right)\right| \rightarrow min \end{cases} \quad (2)$$

where α_1 = 5 degrees is the angle threshold, $p'_{ij,0}, p'_{kl,0}$ and n'_{ij} are projection of $p_{ij,0}, p_{kl,0}$ and n_{ij} on a *xy* plane, $S_{ij}.w$ is the width of the surface S_{ij}, which is a short side of a 2D minimum bounding box (*mbb*) of the points of the surface S_{ij} projected on its fitting surface, and the distance is a sign distance.

Subsequently, candidate points (PB_i) of the beam B_i are given in Eq. 3. The first condition is to obtain the points between two columns. The second condition is based on the beam's width no larger than the column's width. Moreover, when the surface S_{kl} is not available, which implies the wall supports the beam, the distance $d\left(p_{kl,0}, S_{ij}(p_{ij,0}, n_{ij})\right)$ is set as an infinity, and only $S_{ij}.w$ is used in the second condition.

$$p_i \in PB_i \ if \begin{cases} 0 \leq d\left(p_i, S_{ij}(p_{ij,0}, n_{ij})\right) \ \wedge \ d\left(p_i, S_{ij}(p_{ij,0}, n_{ij})\right) \leq d\left(p_{kl,0}, S_{ij}(p_{ij,0}, n_{ij})\right) \\ \left| d(p_i, L_{jl})\right| \leq \max(S_{ij}.w, S_{kl}.w) + tol \end{cases} \quad (3)$$

where $d(p_i, L_{jl})$ is a sign distance from the points p_i to the line $L_{ij} = p'_{ij,0}\, p'_{kl,0}$, $S_{kl}.w$ is the width of the surface S_{kl}, and *tol* = 0.1m is a tolerance compensating data errors or imperfect structures.



Step 1: Beam extraction based column & wall *Step 2: Filter-base voxel and surface*

Figure 5**:** Illustration of beam extraction

Step 2: Similar *Step 2* of the column extraction, VRG is to extract points of planar surfaces, CSC algorithm is to determine the final surfaces and SbF filters outlier points (Fig. 5). Notably, the initial surface for CSC algorithm is the bottom surface defined as the largest surface with the smallest angle between its normal vector and a unit vector of the oz axis (n_z = [0, 0, 1]).

4. Structure Inspection

This section is to inspect main structural components (ceiling slabs, columns and beams) in terms of deformations. A deformation is defined as a distance between the point cloud of the components' surfaces to the reference surface (S_{ref}) considering as the un-deformed surface.

4.1 Ceiling slab deformation

In concrete buildings, ceiling slabs are often supported by the beams and/or shear walls, and the slabs' edges are fixed. As such, intersection lines (L_{int}) between the slab's surface and

supported element's surfaces (S_{SE}) are used to determine the reference surface (S_{ref}). Moreover, S_{SE} is perpendicular to the slab's surface. Thus, the algorithm identifies L_{int} to determine S_{ref} and computes slab's deformations.

For each slab $Slab_i$, the algorithm starts with boundary patches ($\psi_{ext,ij} \in c_i \subset Slab_k$) to retrieve adjacent cells c_j sharing an edge with c_i. Notably, c_j does not possess any patch of the slab $Slab_i$. Next, candidate points p_i of $S_{SE,i}$ are given in Eq. 4. Subsequently, VRG is employed to segment $p_i \in S_{SE,i}$, and the final surface of $S_{SE,i}$ is the closest surface , perpendicular to $\psi_{ext,ij}$.

$$p_i = \begin{cases} (p_{ci} \in c_i) \wedge (p_{cj} \in c_j) \; where \; p_{ci} \notin Slab_i \wedge p_{cj} \notin Slab_i \\ \left| d\left(p_{SSEi}, \psi_{ij.boundext}(p_{ij,0}, n_{ij})\right)\right| \leq d_1 \end{cases} \quad (4)$$

where d_1 = 0.3m is the distance threshold.

An intersection line segment $L_{int,i}$ between $\psi_{ext,ij}$ and $S_{SE,i}$ is determined, and the middle point of $L_{int,i}$ is considered as the edge point. The reference surface $S_{ref,i}(p_{ref,i}, n_{ref,i})$ of $Slab_i$ is fitted through all edge points using PCA and the deformation of $Slab_i$ is expressed in Eq. 5.

$$d\left(p_i \in Slab_i, S_{ref,i}(p_{ref,i}, n_{ref,i}\right) = \frac{(x_i - x_0)n.x + (y_i - y_0)n.y + (z_i - z_0)n.z}{\sqrt{n^2.x + n^2.y + n^2.z}} \quad (5)$$

where $p_{ref,i} = (x_0, y_0, z_0)$ and $n_{ref,i} = (n.x, n.y, n.z)$ is the centroid of the edge points and the normal vector, and $p_i = (x_i, y_i, z_i) \in Slab_k$.

4.2 Column verticality

The column verticality is measured as out-of-plumbness of the column against a perfectly vertical surface (called the un-deformed surface, S_{ref}) through of the column top. The surface S_{ref} is determined through an intersection line L_{int} between the column's surface and the surface of the connected component (S_{CC}). The column often supports the beams (Case 1) and/or the ceiling slabs (Case 2), which can be automatically identify based on results from *Modulus 3*. The algorithm starts to determine L_{int} and then compute verticality.

For each surface S_{ij} of the column Col_i, PCA is employed to estimate unit vectors (n_{ij} – a normal vector, t_{ij} – a small tangent vector, tt_{ij} – a large tangent vector) while the 2D *mbb* is used to determine its width ($S_{ij}.w$) and height ($S_{ij}.h$).

For *Case 1*, from *Modulus 3*, if a beam B_k connecting to the column Col_i is available, the bottom surface S_{kl} of the beam B_k is retrieved. A sub-data set of S_{kl} given in Eq. 6 is representing $S_{CC,kl}$.

$$p_i \in S_{kl} \rightarrow p_{SCC,ij} \; if \; 0 \leq d\left(p_i, S_{ij}(p_{ij,0}, n_{ij})\right) \wedge d\left(p_i, S_{ij}(p_{ij,0}, n_{ij})\right) \leq S_{ij}.w \quad (6)$$

where $p_{ij,0}$ is a centroid of the points of the surface S_{ij}, and the distance here is a sign distance.

In *Case 2*, if a ceiling slab $Slab_k$ connects the column Col_i, only a subset of the slab expressed in Eq. 7 is considered to represent $S_{CC,kl}$.

$$p_i \in Slab_k \rightarrow \; p_{SCC,ij} if \begin{cases} 0 \leq d\left(p_i, S_{ij}(p_{ij,0}, n_{ij})\right) \wedge d\left(p_i, S_{ij}(p_{ij,0}, n_{ij})\right) \leq S_{ij}.w \\ \left| d\left(p_i, S_{ij}^t(p_{ij,0}, t_{ij})\right)\right| \leq 0.5 S_{ij}.w \end{cases} \quad (7)$$

Subsequently, the intersection line, $L_{int,jl}(p_{int,jl}, t_{int,jl})$ is determined from the surface S_{ij} and $S_{CC,kl}$. Next, the reference surface $S_{ref,ij}$ is defined by a normal vector $n_{ref,ij}$ as a cross product of $t_{int,jl}$ and n_z. Finally, the verticality is distances $d(p_i \in S_{ij}, S_{ref,ij}(p_{int,jl}, n_{ref,ij})$ as given in Eq. 5.

4.3 Beam deformation

In concrete buildings, the deformation of a beam can express as the distance between the bottom surface S_{ij} of the beam B_i and the reference surface $S_{ref,i}$ through the beam supports. That is because the beam supports are assumed to be fixed. Moreover, as mentioned above, the end beams are supported by column (*Case 1*) and/or a shear wall (*Case 2*), which is automatically identified based on results of beam extraction in *Modulus 3*. For the bottom surface S_i of the beam B_i, unit vectors (n_{ij} – a normal vector, t_{ij} – a small tangent vector, tt_{ij} – a large tangent vector) and the surface dimensions ($S_{ij}.w$ - width and $S_{ij}.h$ - height) are respectively estimated by PCA and the 2D *mbb*. Notably, the direction of n_{ij} is the same to one of n_z.

For *Case 1*, the surface S_{kl} of the column Col_k connecting to the beam B_i is extracted, and then a sub-data set of the surface S_{kl} is to represent $S_{SE,ij}$, which is expressed in Eq. 8.

$$p_i \in S_{kl} \rightarrow p_{SSE,kl}\ if - S_{ij}.w \leq d\left(p_i, S_{ij}(p_{ij,0}, n_{ij})\right) \wedge d\left(p_i, S_{ij}(p_{ij,0}, n_{ij})\right) \leq 0 \quad (8)$$

where $d\left(p_i, S_{ij}(p_{ij,0}, n_{ij})\right)$ is a sign distance

In *Case 2*, the walls extracted from Modulus 1 closed to the end of the beam B_i is extracted. A sub-data set of the wall $Wall_k$ determined based on Eq.9, is used to determine $S_{SE,k}$.

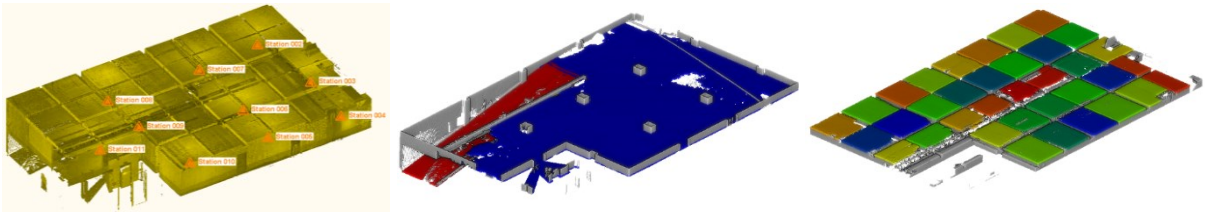
$$p_i \in Wall_k \rightarrow p_{SCE,k} if \begin{cases} -S_{ij}.w \leq d\left(p_i, S_i(p_{ij,0}, n_{ij})\right) \wedge d\left(p_i, S_{ij}(p_{ij,0}, n_{ij})\right) \leq 0 \\ \left|d\left(p_i, S_{ij}^t(p_{ij,0}, t_{ij})\right)\right| \leq 0.5 S_{ij}.w \end{cases} \quad (9)$$

The intersection line, $L_{int,jl}(p_{int,jl}, t_{int,jl})$ is determined from the surface S_{ij} and $S_{SE,k}$, and the reference surface $S_{ref,ij}$ is the surface through two intersection lines at the ends, in which the direction of $n_{ref,ij}$ is the same n_z. Finally, the distance $d(p_i \in S_{ij}, S_{ref,ij}(p_{int,jl}, n_{ref,ij}))$ given in Eq. 5 is the beam deformation.

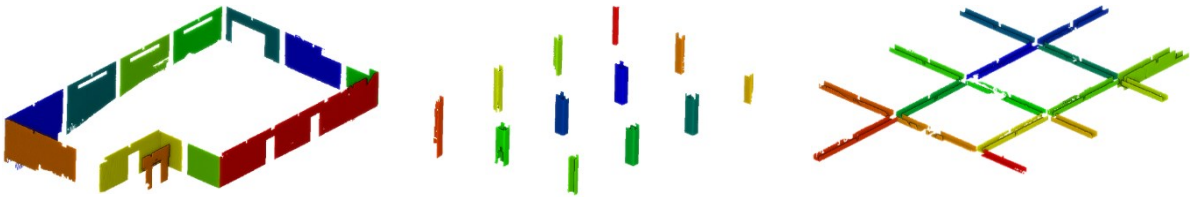
5. Experiments and Results

To demonstrate the proposed method, a ground storey of a office building on Pham Ngu Lao st., Vietnam is selected. The storey is about 18.5m wide x 29.5m long x 3.45m high, and was scanned by a Trimble TX8 with a maximum scanning range at 120m and an angular accuracy of 8μrad in both vertical and horizontal (Trimble, 2020b). A point spacing of 11.3mm at a range of 30m and a total of 11 scanning stations was established to capture an interior storey with a maximize data coverage (Fig. 6a). The point clouds were registered by the Trimble RealWork software v11.2 (Trimble, 2020a) with the registration error about 1.57mm. Finally, 23.5 million points with x-, y- and z- coordinates was exported as input data for the proposed method. Notably, parts of a MEP system were installed, which obstructs to capture surfaces of several structural elements.

A set of parameters for the proposed method as following. For 2D cell decomposition, *cell_size* =1.0m and *min_ptc* = 10 are selected to ensure at least one cell representing the smallest slab by 2mx2m. The bandwidth of 0.2m is set for patch extraction by KDE, which allows to separate two surfaces of the thinnest component. Moreover, the thresholds α_0 = 5 degrees, d_0 = 10mm, r_0 = 10mm and *voxel_size* = 0.1m are selected for CpRG and VRG. Although the surfaces of the building components are almost perpendicular, the small angle threshold is selected to prevent points of the MEP components including in the ceiling segments. Notably, the distance and residual thresholds can adjust based on data errors and the surface roughness. Finally, the selected *voxel_size* is to ensure one voxel can representing a surface of the column and beam.



a) Point cloud of a storey b) Extracted floor c) Extracted ceiling slabs

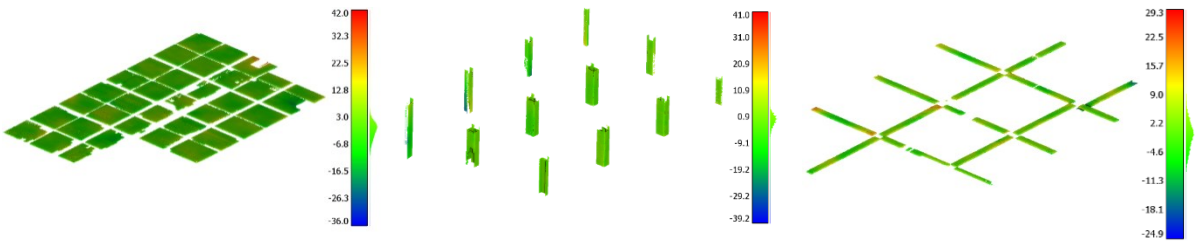


d) Wall e) Column f) Primary beam

Figure 6: Point cloud of a storey from an internal scan and resulted component extraction

As a point cloud of one storey captured used as an input data for the proposed method, patches belonging to the floor and ceiling are mostly the first and last patches of a cell in the vertical direction, respectively. Thus, the first and last patches are respectively set as seeding patches for *Step 2.1* in floor and ceiling extraction, while all remaining patches use for Step 2.2 and 2.3. Once the points are assigned for the components, they are immediately deactivated, and only remaining points are used in subsequent steps. Results of the component extraction are shown in Fig. 6b-e. That can be seen all surfaces of the components (floor and ceiling slabs, columns and beams) are successfully extracted. However, in future work, additional experiemental tests and the quantitative elevation including the level of locational deviation, shape similarity and positional accuracy are implemeting to give detailed the perfromance of the proposed method.

Additionally, as only parts of the data sets are used in extracting the components, it shows that the proposed method is efficiently accommodates a large data set. For example, with a current data set, the processing time is 412.1 seconds including 166.9 seconds for floor and ceiling, 58.3 seconds for wall, 41.1 seconds for column and 145.8 seconds for beam. An executing time of the beam is larger than other components because the voxel size of 0.1m is set for VRG. This value can be adjusted based on the actual component size rather than a fixed value. This performance is based on an implementation of the proposed method in MATLAB 2019b (2019b) and processing on Dell Precision Workstation with a main system configuration as follows: Intel(R) Xeon(R) W-2123 CPU @ 3.6GHz with 32GB RAM.



a) Ceiling slab deformation b) Column verticality c) Beam deformation

Figure 7: Results of building component inspection through deformation

Resulted inspection of the ceiling slabs, columns and beams are shown in Fig. 7. The slab deformations vary in a range from -36.00mm to 42.00mm, but deformations of 99.0% points are in a range a mean deformation (μ) ± 3 times of a standard deviation (σ) ($\mu \pm 3\sigma \equiv$ [-13.62mm, 10.54mm]. Large hogging deformations occurs around the lift where the point cloud of the lift's wall does not fully eliminate out of the slab segment. For the columns, the report shows that 98.0% of the points having the verticality varies in a range from -15.4mm to 14.2mm ($\mu = -0.62$mm and $\sigma = 4.94$mm), where the maximum and minimum verticality are -39.2mm and 41.0mm (Fig. 6b). Similarly, the beam deformations also change from -24.9mm to 29.3mm, but about 98.9% of the point deformations is in a range -13.35mm to 13.35mm ($\mu = 0.0$mm and $\sigma = 4.45$mm). In several beams, the large deformations are found in edges of the bottom surfaces because of an over-segmentation when extracting the points of the bottom surface. Thus, these points must be filtered to avoid miss-leading deformation report.

6. Conclusions

This paper presents an efficient, automatic method to extract structural components of a construction project of a reinforced concrete building. In structural component extraction, the proposed method consists of 3 consecutive moduli to extract the building components in a sequent order: floors, ceilings and wall, columns and beams, in which both spatial information of a point cloud and contextual knowledge about the structures are used. One of advantages of this methods is to roughly extract potential data points of components (floors, ceilings, columns and beams) before using cell-patch and voxel-based region growing to segment final surfaces of the components. As such, the proposed method only processes less complex, small sub-data set to extract the components, which makes the method efficient to a big data encountered in practice. This can be seen through a report of an experimental test on a 23.5 million points of one building storey, in which the proposed method successes to extract all surfaces of the building components with an executing time about 412.1 seconds. Moreover, by implementing the fundamental design of the structure, the component's boundaries can be automatically identified to establish un-deformed surfaces of the components for deformation measurement. In future work, the proposed method is going to test different types, layouts of the buildings and quantitative elevation strategy is implementing to give a detailed evaluation report. In addition, although the proposed method is developed for structure inspection, it could be extended for as-built BIM reconstructions.

Acknowledgements

This work was funded by the generous support of the European Commission through H2020 MSCA-IF, "*BridgeScan: Laser Scanning for Automatic Bridge Assessment*", Grant 799149. The first author is graceful for this support. The authors also thank Dat Hop Company Limited and Ceotic., JSC for their providing the laser scanning data.

References

Bosché, F. (2010). Automated recognition of 3D CAD model objects in laser scans and calculation of as-built dimensions for dimensional compliance control in construction. *Advanced Engineering Informatics,* 24**,** pp.107–118.

Burati Jr, J. & Farrington, J. (1987). Construction industry institute–Costs of quality deviations in design and construction. *Clemson University, Source Document,* 29.

Dimitrov, A. & Golparvar-Fard, M. (2015). Segmentation of building point cloud models including detailed architectural/structural features and MEP systems. *Automation in Construction,* 51**,** pp.32–45.

Kim, M.-K., Sohn, H. & Chang, C.-C. (2014). Automated dimensional quality assessment of precast concrete panels using terrestrial laser scanning. *Automation in Construction,* 45**,** pp.163–177.

Laefer, D. F. & Truong-Hong, L. (2017). Toward automatic generation of 3D steel structures for building information modelling. *Automation in Construction,* 74**,** pp.66–77.

Maalek, R., Lichti, D. D. & Ruwanpura, J. Y. (2019). Automatic recognition of common structural elements from point clouds for automated progress monitoring and dimensional quality control in reinforced concrete construction. *Remote Sensing,* 11**,** 1102.

Mathworks (2019b). MATLAB Function Reference. 2019b ed.

Rabbani, T., Heuvel, F. V. D. & Vosselmann, G. (2006). Segmentation of point clouds using smoothness constraint. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences,* 36**,** pp.248–253.

Trimble. (2020a). *Trimble RealWorks v11.2* [Online]. Trimble. Available: https://geospatial.trimble.com/products-and-solutions/trimble-realworks [Accessed 01/15 2020].

Trimble. (2020b). *Trimble TX8 LASER SCANNER* [Online]. Trimble. Available: https://geospatial.trimble.com/products-and-solutions/trimble-tx8 [Accessed 01/15 2020].

Truong-Hong, L., Chen, S., Cao, V. L. & Laefer, D. F. (2018). Automatic bridge deck damage using low cost UAV-based images.

Truong-Hong, L. & Lindenbergh, R. (2020). Quantitative assessment of structural components for construction management using laser scanning data. FIG Working Week 2020, 10-14 May 2020 Amsterdam, Netherlands. 15.

Vo, A.-V., Truong-Hong, L., Laefer, D. F. & Bertolotto, M. (2015). Octree-based region growing for point cloud segmentation. *ISPRS J Photogramm Remote Sens,* 104**,** pp.88–100.

Wang, Q., Cheng, J. C. P. & Sohn, H. (2017). Automated estimation of reinforced precast concrete rebar positions using colored laser scan data. *Computer-Aided Civil and Infrastructure Engineering,* 32**,** pp.787–802.