SoftwareX 18 (2022) 101099

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

Cloud2FEM: A finite element mesh generator based on point clouds of existing/historical structures

Giovanni Castellazzi *, Nicolò Lo Presti, Antonio Maria D'Altri, Stefano de Miranda

Department of Civil, Chemical, Environmental and Materials Engineering (DICAM), University of Bologna, Viale del Risorgimento 2, Bologna 40136, Italy

ARTICLE INFO

Article history:
Received 14 February 2022
Received in revised form 20 April 2022
Accepted 3 May 2022

Keywords:
Point cloud
FE modelling
Mesh generation
Cultural heritage structures
Voxel
Masonry

ABSTRACT

Nowadays, the common output of surveying activities on existing/historical structures consists of dense point clouds. However, the direct and automatic exploitation of point clouds for structural purposes, i.e. to generate finite element models, is still very limited. In this framework, the Cloud2FEM software supplies an automatic finite element mesh generator based on point clouds of existing/historical structures. Cloud2FEM is based on open-source Python libraries with graphical interface. The point cloud is initially sliced along with the vertical direction. Then, closed polygons are recognized on each slice and stacked vertically thanks to the use of voxels. The voxelized volume is exported into 3D solid hexahedron-based finite element meshes. Suitable graphical tools are developed to help the user adjusting local potential criticalities in the slices, also when partial information is missing in the points cloud. An illustrative example is given to highlight the Cloud2FEM potentialities.

Code metadata

Current code version	v1.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-22-00047
Permanent link to reproducible capsule	None
Legal code license	GNU General Public License (GPL) version 3
Code versioning system used	None
Software code languages, tools and services used	Python
Compilation requirements, operating environments and dependencies	PyQt5, PyQtGraph, VisPy, NumPy, pyntcloud, Shapely, ezdxf.
If available, link to developer documentation/manual	None
Support email for questions	giovanni.castellazzi@unibo.it

1. Motivation and significance

The structural analysis of existing/historical structures is a challenging task. One main reason consists in the geometrical complexity which typically characterize these structures, that lead to non-trivial tasks for the geometrical modelling in the framework of the Finite Element Method (FEM). A promising support could come from modern automatic survey techniques (e.g. laser scanning and close-range photogrammetry) which produce dense point clouds [1] of the building under study. However, the direct exploitation of point clouds for structural purposes, i.e. to generate Finite Element (FE) models, is still very limited. Indeed, only few attempts concerning historic facades [2], timber structures [3], minarets [4], and pulpits [5], have been lately developed. Typically, the standard process consists in the manual creation of a CAD solid model and its discretization with solid FEs, even if this usually leads to errors in the FE meshing procedures due to geometric imprecisions and tolerances.

In this framework, Castellazzi et al. [6,7] proposed a workflow to directly and semi-automatically exploit dense point clouds to generate FE meshes. In particular, the point cloud is initially sliced along with the vertical direction. Hence, closed polygons are recognized on each slice and stacked vertically thanks to the use of voxel concept. The voxelized volume is then exported into robust and conforming 3D solid hexahedron-based FE meshes. This workflow guarantees a remarkable user-time saving with respect to 3D CAD standard modelling procedures, as well as a considerable robustness of the generated mesh. These aspects have been underlined in [8], where nonlinear static analysis have

* Corresponding author.
E-mail address: giovanni.castellazzi@unibo.it (Giovanni Castellazzi).

https://doi.org/10.1016/j.softx.2022.101099

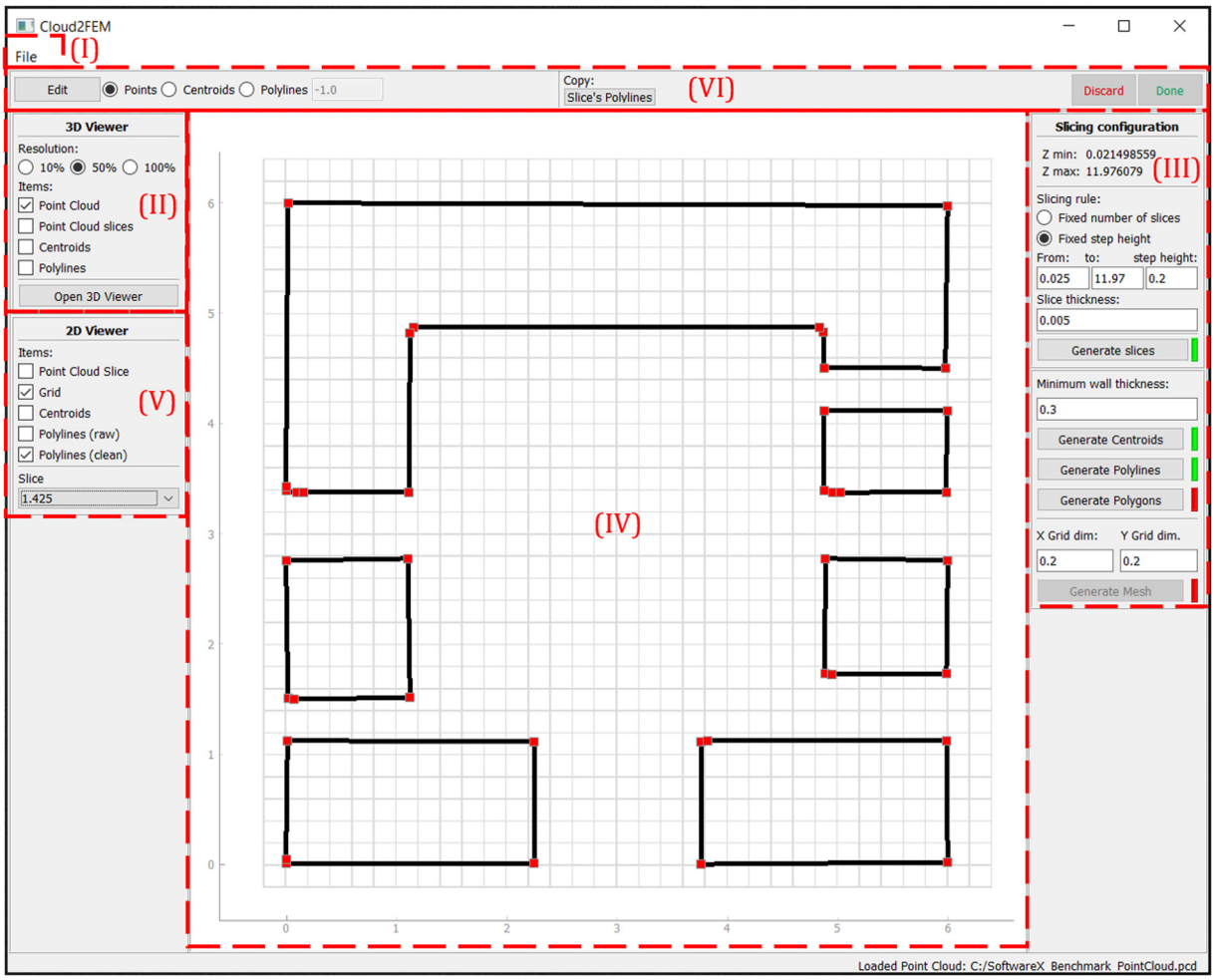

Fig. 1. Cloud2FEM main graphical user interface: (I) file menu, (II) 3D Viewer panel, (III) cloud/mesh processing panel, (IV) plot area, (V) 2D Viewer panel, (VI) editing bar.

been successfully conducted on the generated mesh of a historical fortress. In particular, this workflow demonstrated to be efficient also in presence of horizontal decks and masonry vaults in the structure, as discussed in [7].

In this paper, an easy-to-use open-source software (named Cloud2FEM) which efficiently accomplishes the point cloud-to-numerical model procedure proposed in [6,7] is released. The workflow proposed in [6,7], initially composed of different scripts written in different code languages, has been herein straightforwardly implemented using the Python language. In particular, the Cloud2FEM software has been developed with a user-friendly graphical interface. Accordingly, this allows an agile and robust seamless user experience and easy improvements of the code. Therefore, non-skilled users are able to approach the software and transform a point cloud (input) into a solid FE model (output), allowing a widespread adoption of this approach. Additionally, the graphical interface favoured the implementation of CAD-like tools which can be used by the user to adjust local potential criticalities in the slices, also when partial information is missing in the point clouds.

2. Software description

The main graphical user interface of the software Cloud2FEM is shown in Fig. 1. On the top (I), the file menu contains buttons to load point clouds (the input data of this software), save a project, load a previously saved project, and export data to be used into CAD environments (i.e. .dxf slices) or FE simulations (i.e. FE solid mesh). On the left (II), the 3D Viewer section contains buttons to open a separated window where the selected quantities can be explored in 3D.

The right panel (III) contains, starting from the top, the extreme z coordinates of the loaded point cloud, a section to specify the slicing modality, buttons to perform all the steps needed to generate a FE model. Red and green indicators beside each button denote if a certain step still needs to be performed or not, respectively. Most of the data generated through the steps in the right panel can be visualized in the central plot area (IV), which occupies most of the space of the interface. The 2D Viewer section on the left (V) can be utilized to choose the data to be shown, for any slice. The edit button on the top bar (VI) allows to enter the edit mode for the current slice and for the selected data type. An example of slice processing is shown in Fig. 2.

On each slice, local modifications can be performed with the help of various suitable ad-hoc tools (for both points and polylines, e.g. draw, join, remove, add, move, offset, etc.), that can be activated via keyboard shortcuts. The top bar (VI) contains also buttons to save or discard the changes made in the edit mode, as well as a copy button that opens a separated window used to copy data from one slice to others. Finally, the button "Generate mesh" positioned at the bottom of the right panel (III) generates the matrix of nodal coordinates and the connectivity matrix of the FEs, i.e. the FE mesh, which can be exported from the file menu (I). In this particular case, the mesh is exported into

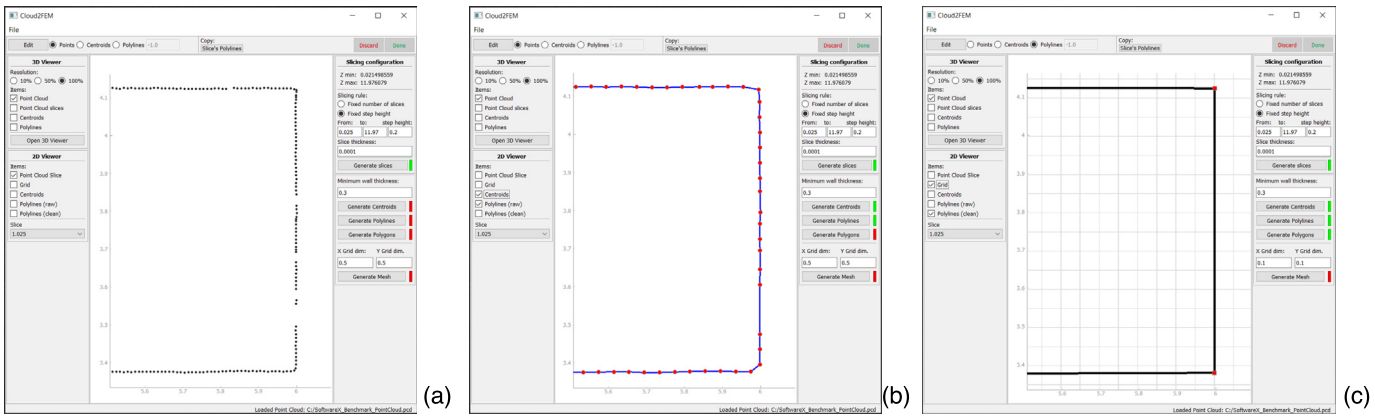

Fig. 2. Example of slice processing through Cloud2FEM: (a) slice of the raw point cloud, (b) derived centroids and polylines before simplification, (c) final clean geometry ready to be used in the mesh generation step.

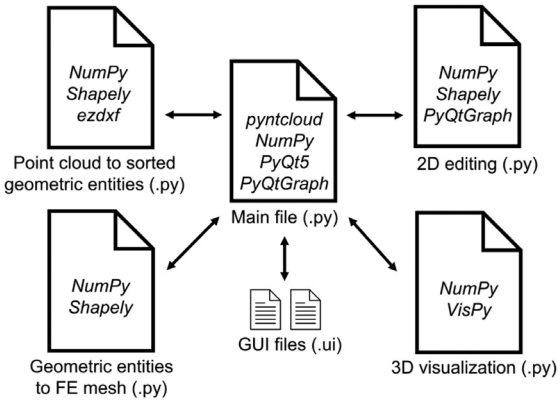

Fig. 3. Cloud2FEM software architecture and libraries involved.

the .inp format [9], which can be visualized also by open-source software packages, see e.g. FreeCAD [10]. Anyway, the matrix of nodal coordinates and the connectivity matrix of the FEs can be found unencrypted in the .inp file (text format), and so they can be used in any available FE software.

2.1. Software architecture

Cloud2FEM is a Python-based software [11]. Its graphics (see e.g. Fig. 1) is realized with the help of the PyQt5 library [12], supplemented by the PyQtGraph library [13] for 2D plotting and editing, and by the VisPy [14] library for 3D visualization. On the back-end, various open source libraries are employed. At the base, NumPy [15] is widely utilized for array computing. The input point cloud data in the .pcd or .ply format is converted into a Python data type by means of the pyntcloud library [16]. The Shapely package [17] is adopted to manipulate 2D geometric entities with ease, and ezdxf [18] is exploited to export 2D slices in the .dxf format. The code consists of a main file, two .ui files where the graphic entities are stored, and four thematic modules with the functions utilized for 3D visualization, conversion of a point cloud in a set of sorted geometric entities, conversion of geometric entities in a FE mesh, manual editing of points and polylines. The code structure and the libraries involved are shown in Fig. 3.

2.2. Software functionalities

The main functionalities of Cloud2FEM are here listed:

- Loading a point cloud in .pcd or .ply formats and visualizing it with the 3D viewer. Since the 3D visualization of a large point cloud can be computationally demanding, the software uses a graphical accelerated tool to support real large dataset visualization. Nevertheless, the user can choose between three possible predefined resolutions (10, 50 and 100%) to get a smoother interaction.
- Visualizing the extreme $[Z_{min}, Z_{max}]$ coordinates of the loaded point cloud.
- Slicing of the point cloud. The user can choose between two slicing rules, i.e. fixed number of slices or fixed step height. The function activated by the "Generate slices" button extracts slices from the loaded point cloud, according to the specified slicing parameters. Every slice, named by its Z coordinate, can be selected from the drop-down list on the left panel and visualized in the 2D viewer. The whole set of slices can be visualized in the 3D viewer by checking the corresponding checkbox on the left panel.
- Simplifying and ordering the slices. Every slice, consisting of an unordered array of points, is processed through the algorithm called by the "Generate Centroids" button to get a new smaller array of ordered points that can be used to efficiently describe the geometry of the slice. The centroids can be visualized in the 2D viewer by checking the corresponding checkbox in the left panel.
- Generating polylines. The centroids data generated by the software in a previous step is utilized to get a set of raw polylines. These are automatically simplified to get a cleaner and lightweight geometric representation. Both the raw and the clean polylines can be visualized in the 2D viewer by checking the corresponding checkbox in the left panel.
- Generating polygons. A more descriptive data type is automatically derived from the polylines obtained in a previous step. Each slice is described by means of polygons, made of inner and outer bounds. This geometric representation can be exported in the .dxf file format.
- Generating the mesh. A FE mesh is automatically generated from the polygons data. The X and Y dimensions of the voxels are set according to the grid visible in the 2D viewer, while the Z dimension depends on the adopted slicing parameters. The output can be exported in a text file format organized according to the Abaqus notation (.inp) [9].
- Saving and opening a project. The data handled by the software can be stored and retrieved to allow discontinuous work sessions. Furthermore, since the data is stored using the built-in Python shelve module, advanced users can easily exploit the data in other Python-based software.

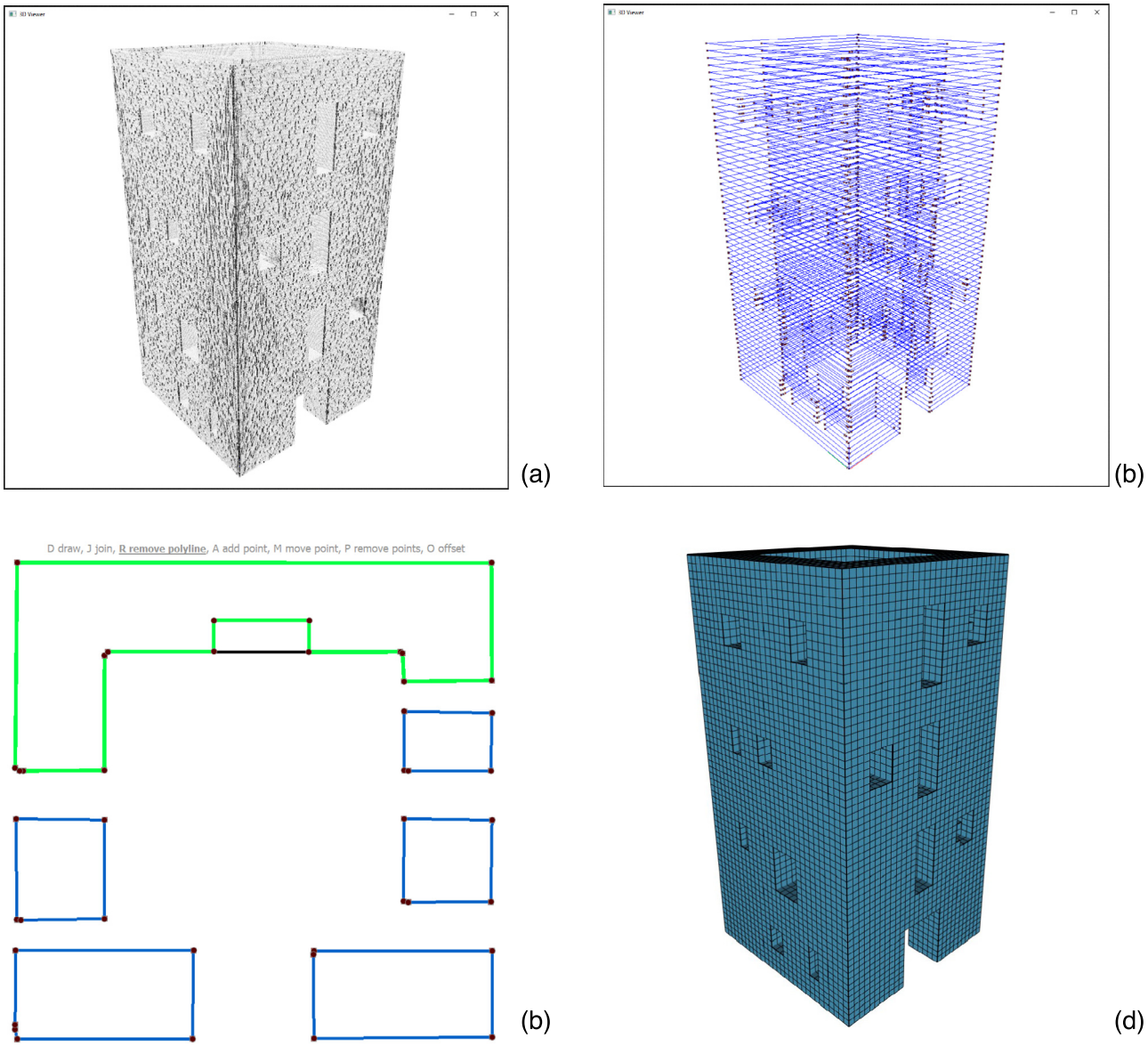

Fig. 4. Example of generation of a FE mesh based on a point cloud of a benchmark structure by means of the Cloud2FEM software: (a) point cloud used as input visualized at 100% resolution, (b) slicing of the point cloud, (c) slice example with local modification, (d) FE mesh (output).

- Editing of points and centroids. To eliminate criticalities in the slices, points and centroids can be removed individually or in bulk with graphic tools in the 2D viewer.
- Editing of polylines. A set of tools in the 2D viewer allows for an advanced graphic editing of the polylines. Vertices can be added, removed or moved, polylines can be joined, removed or drawn from scratch if geometric data is missing from the original point cloud.
- Copy of the polylines of a slice. The polylines of a slice can be copied to other to speed up the editing process.

3. Illustrative example

An illustrative example of the automated generation of a conforming FE mesh based on a point cloud of a pseudo-actual benchmark structure by means of the Cloud2FEM software is shown in Fig. 4. The structure has a height equal to 12 m, a square base with a length of 6 m and its walls are roughly 1 m thick. Fig. 4(a) shows the point cloud used as input (5,977,514 points). The representation of the initial geometry of the benchmark through slices composed of closed polygons is shown in Fig. 4(b). An example of slice is shown in Fig. 4(c), where a local modification through ad-hoc tools is also highlighted. Finally, Fig. 4(d) shows the FE mesh (composed of 38,882 nodes and 31,638 8-node hexahedral FEs) generated through the Cloud2FEM software, i.e. the main output of the software, ready to be used for structural analysis purposes. It should be noted that the visualization in Fig. 4(d) has been obtained through the open-source software FreeCAD [10].

The parameters adopted in the example in Fig. 4 are:

- Slicing with a fixed step height of 0.2 m, from 0.025 m to 11.97 m.
- Slice thickness S_t equal to 0.005 m.
- Minimum wall thickness M_{wt} equal to 0.3 m.
- X and Y grid dimensions set equal to 0.2 m.

It should be highlighted that these parameters are here listed to guarantee the reproducibility of the model in Fig. 4. For the sake of example, the influence of the voxel size on the structural

performance of voxel-based numerical models has been assessed in [19]. Accordingly, the choice of the voxel dimensions can be conducted in agreement with the suggestions given in [6,7,19].

Moreover, S_t refers to the tolerance in the Z direction, i.e. a point P belongs to the slice Z_i if its Z coordinate Z_P is included between the limits $Z_i - \frac{S_t}{2} \leq Z_P \leq Z_i + \frac{S_t}{2}$. At first, the value of S_t can be typically set equal to 5 mm and then, if needed, adjusted depending on the density of the actual point cloud. Furthermore, M_{wt} refers to a tolerance used by the software to correctly identify distinct portions in a slice, and it can be set depending on the actual minimum wall thickness or, if smaller, on dimension of the smaller opening measurable in the structure.

Finally, it should be highlighted that point clouds made of up to 150 million points have been easily manipulated with Cloud2FEM (on a commercial laptop equipped with 16 GB of RAM and a dedicated GPU).

4. Impact

The impact of the Cloud2FEM software on common practices in structural analysis of cultural heritage buildings appears significant. First of all, Cloud2FEM allows non-skilled users to generate solid and robust FE models of existing/historical buildings starting from raw dense point clouds of any complexity and any level of completeness. Indeed, the software herein discussed guarantees its usability also in case of non-comprehensive point clouds, e.g. point clouds surveyed only on the external building surfaces, which are the most common case encountered in practice. This allows the direct and automatic exploitation of point clouds for structural purposes, i.e. for the generation of voxel-based FE models which demonstrated to be particularly appealing for both model updating [20] and seismic assessments,8. Therefore, this software represents a generalization of the procedures used in [6–8], characterized by the possibility to deal with a larger range of cases and by a smoother user experience.

Cloud2FEM usage appears preferable with respect to standard 3D CAD-based procedures, given its speediness and robustness. Indeed, it always guarantees the generation of robust conforming solid FE meshes, and it allows to save user-time with respect 3D CAD approaches [6]. In particular, it requires no CAD skills from the user, who needs basically no training to be able to use Cloud2FEM.

Therefore, Cloud2FEM could become the starting point for advanced structural assessments of cultural heritage structures [4,8,21–24], used by both research groups and consultancy professional teams. For example, it will be thoroughly used in the H2020-funded HOLAHERIS project [25]. The simplicity of the Cloud2FEM usage also encourages its adoption in student and teaching activities (e.g. theses, homework, etc.).

5. Conclusions

The Cloud2FEM software supplies a FE mesh generator which receives as input the point cloud of an existing/historical structure, and gives as output the solid FE model of the structure. Cloud2FEM is based on open-source Python libraries with graphical interface, which demonstrated to be particularly user-friendly. The data processing follows the initial slicing of the point cloud along with the vertical direction, the automatic recognition of closed polygons on each slice, and the slice stacking using the concept of voxel. Finally, the voxelized volume is exported into 3D solid hexahedron-based FE meshes ready to be used for structural purposes.

Ad-hoc graphical tools have been developed to help the user adjusting local potential criticalities in the slices, also when partial information is missing in the point cloud. Accordingly, Cloud2FEM allows the generation of solid FE models of existing/historical buildings starting from point clouds of any level of completeness, i.e. also in the case of non-comprehensive point clouds, which is the most common case encountered in practice.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 101029792 (HOLAHERIS project, "A holistic structural analysis method for cultural heritage structures conservation").

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.softx.2022.101099.

References

[1] Riveiro B, DeJong MJ, Conde B. Automated processing of large point clouds for structural health monitoring of masonry arch bridges. Autom Constr 2016;72:258–68. http://dx.doi.org/10.1016/j.autcon.2016.02.009.
[2] Truong-Hong L, Laefer DF. Validating computational models from laser scanning data for historic facades. J Test Eval 2013;41(3):481–96. http://dx.doi.org/10.1520/jte20120243.
[3] Armesto J, Lubowiecka I, Ordóñez C, Rial FI. FEM modeling of structures based on close range digital photogrammetry. Autom Constr 2009;18(5):559–69. http://dx.doi.org/10.1016/j.autcon.2008.11.006.
[4] Korumaz M, Betti M, Conti A, Tucci G, Bartoli G, Bonora V, et al. An integrated terrestrial laser scanner (TLS), deviation analysis (DA) and finite element (FE) approach for health assessment of historical structures, a minaret case study. Eng Struct 2017;153:224–38. http://dx.doi.org/10.1016/j.engstruct.2017.10.026.
[5] Bartoli G, Betti M, Bonora V, Conti A, Fiorini L, Kovacevic VC, et al. From TLS data to FE model: A workflow for studying the dynamic behavior of the pulpit by Giovanni Pisano in Pistoia (Italy). Proc Struct Integ 2020;29:55–62.
[6] Castellazzi G, D'Altri AM, Bitelli G, Selvaggi I, Lambertini A. From laser scanning to finite element analysis of complex buildings by using a semi-automatic procedure. Sensors 2015;15(8):18360–80. http://dx.doi.org/10.3390/s150818360.
[7] Castellazzi G, D'Altri AM, de Miranda S, Ubertini F. An innovative numerical modeling strategy for the structural analysis of historical monumental buildings. Eng Struct 2017;132:229–48. http://dx.doi.org/10.1016/j.engstruct.2016.11.032.
[8] Degli Abbati S, D'Altri AM, Ottonelli D, Castellazzi G, Cattari S, de Miranda S, et al. Seismic assessment of interacting structural units in complex historic masonry constructions by nonlinear static analyses. Comput Struct 2019;213:51–71.
[9] Abaqus®. Theory manual, version 6.20. 2020.
[10] Riegel J, Mayer W, van Havre Y. FreeCAD. 2016.
[11] Python language reference, version 3.8.5. Python Software Foundation; 2020, [Online]. Availablewww.python.org [Accessed 23 July 2020].
[12] PyQt5 - Python bindings for Qt v5. 2021, [Online]. Available: https://pypi.org/project/PyQt5/ [Accessed 20 October 2021].
[13] PyQtGraph. Scientific graphics and GUI library for Python, v0.11.1. 2021, [Online]. Available: https://pypi.org/project/pyqtgraph/ [Accessed 21 January 2021].
[14] VisPy. Interactive visualization in Python, v0.6.6. 2021, [Online]. Available: https://pypi.org/project/vispy/ [Accessed 21 January 2021].
[15] NumPy. Package for array computing with python, v1.21.2. 2021, [Online]. Available: https://pypi.org/project/numpy/ [Accessed 13 October 2021].
[16] Pyntcloud. Python library for working with 3D point clouds, v0.1.5. 2021, [Online]. Available: https://pypi.org/project/pyntcloud/ [Accessed 20 October 2021].

[17] Shapely. Python package for manipulation and analysis of planar geometric objects, v1.7.1. 2021, [Online]. Available: https://pypi.org/project/Shapely/ [Accessed 21 January 2021].
[18] Ezdxf. A Python package to create/manipulate DXF drawings, v0.15.2. 2021, [Online]. Available: https://pypi.org/project/ezdxf/ [Accessed 23 February 2021].
[19] Castellazzi G, D'Altri AM, de Miranda S, Ubertini F, Bitelli G, Lambertini A, et al. A mesh generation method for historical monumental buildings: An innovative approach. In: Proceedings of the VII european congress on computational methods in applied sciences and engineering. Crete Island; 2016, http://dx.doi.org/10.7712/100016.1823.11948.
[20] Bassoli E, Vincenzi L, D'Altri AM, de Miranda S, Forghieri M, Castellazzi G. Ambient vibration-based finite element model updating of an earthquake-damaged masonry tower. Structural Control and Health Monitoring 2018;25(5):e2150.
[21] Kassotakis N, Sarhosis V, Riveiro B, Conde B, D'Altri AM, Mills J, et al. Three-dimensional discrete element modelling of rubble masonry structures from dense point clouds. Autom Constr 2020;119:103365.
[22] Almac U, Pekmezci I, Ahunbay M. Numerical analysis of historic structural elements using 3D point cloud data. Open Const Build Technol J 2016;10(1).
[23] Pepi C, Cavalagli N, Gusella V, Gioffrè M. An integrated approach for the numerical modeling of severely damaged historic structures: Application to a masonry bridge. Adv Eng Softw 2021;151:102935.
[24] Funari MF, Hajjat AE, Masciotta MG, Oliveira DV, Lourenço B. A parametric scan-to-FEM framework for the digital twin generation of historic masonry structures. Sustainability 2021;13(19):11088.
[25] HOLAHERIS website. 2022, [Online]. Available: https://site.unibo.it/holaheris/en [Accessed 17 May 2022].