Automatic surface extraction and web visualization workflow for large laser scanner point clouds with open-source solutions

Marcello La Guardia [1], Mauro Lo Brutto [2], Mila Koeva [3], Giuseppe Mussumeci [1]

[1] Department of Engineering, University of Messina, 98158 Messina, Italy – (marcello.laguardia, giuseppe.mussumeci)@unime.it
[2] Department of Engineering, University of Palermo, Viale delle Scienze, 90128, Palermo, Italy- mauro.lobrutto@unipa.it
[3] Faculty of Geo-Information Science and Earth Observation, University of Twente, Enschede, The Netherlands - m.n.koeva@utwente.nl

Keywords: Laser scanning, Survey, surface extraction, WebGL, Point cloud, open source.

Abstract

Recent advances in geomatics and 3D surveying have enabled the acquisition of increasingly dense point clouds through both static and mobile laser scanning systems, supporting the rapid digital documentation of built and natural environments. At the same time, the growing diffusion of WebGL technologies has opened new possibilities for the online visualization and dissemination of complex three-dimensional datasets. Within this context, the present study proposes an open-source workflow for the automatic extraction of significant geometric surfaces from laser scanner point clouds and their integration into a web-based visualization framework. The method was developed within a Python-based processing environment and tested on three datasets characterized by different levels of geometric complexity: a regular built environment, an under-construction building environment, and a historical context. The workflow includes point cloud preprocessing, automated segmentation strategies adapted to the geometric complexity of each case, extraction of planar and non-planar elements, polygonal surface generation, mesh construction, and conversion of outputs into lightweight formats suitable for web publication. The final visualization environment combines segmented polygonal models and subsampled point cloud data through open-source WebGL technologies. The results demonstrate that the proposed strategy provides a scalable and flexible solution for the rapid online representation of large laser scanner datasets, supporting surface recognition, low-cost accessibility, and future semantic enrichment within web-based geospatial and Digital Twin applications.

1. Introduction

Recent advances in geomatics have enabled the acquisition of extremely dense point clouds through both static and mobile laser scanning technologies, allowing real-time 3D data collection across diverse environmental contexts (Aminiti et al., 2020; Alsadik and Karam, 2021; Aricò et al., 2023). In particular, the diffusion of mobile laser scanners has provided specialists with a valuable technology capable of capturing both regular and irregular surfaces. These systems significantly reduce survey time while maintaining a sufficient level of accuracy (Gollob et al., 2020), making them increasingly suitable for the documentation of complex environments, opening new possibilities also to low-cost solutions (Masiero et al., 2020). Mobile laser scanner acquisitions have proven particularly effective in historical buildings, archaeological sites, and articulated architectural spaces, where they support rapid and accurate 3D information extraction and can be integrated into broader Scan-to-BIM and digital documentation workflows (Aricò et al., 2024).

Automatic surface extraction and semantic segmentation from point clouds remains a challenging research field, mainly due to the intrinsic characteristics of point cloud datasets, such as noise, gaps, variable density, occlusions, and non-uniform sampling (Huang et al., 2024; Sulzer et al., 2025). These issues become even more relevant when the surveyed environments include both regular geometries, such as walls and floors, and irregular or curved elements, such as vaults, pillars, rock-cut spaces, or deteriorated architectural components.

Recent studies have explored automated and semi-automated approaches for surface extraction and classification from laser scanner data, including segmentation strategies and deep-learning-based methods (Haznedar et al., 2023; La Guardia et al., 2025; Matellon et al., 2026). However, many workflows remain computationally demanding, difficult to generalize across different geometric contexts, or poorly integrated with lightweight web-sharing environments.

At the same time, the rapid development of WebGL technologies has facilitated the visualization of complex 3D environments on the web, including large point clouds acquired through laser scanning (La Guardia et al., 2022).

Open-source JavaScript libraries have significantly improved the possibility of interactively navigating 3D scenes in a browser, while optimization techniques have partially addressed the limitations related to the large size of point cloud datasets (Gaspari et al., 2024). Nevertheless, many available solutions are still primarily focused on visualization and geometric inspection rather than on the direct integration of automated processing outputs, such as segmented surfaces, extracted primitives, or simplified polygonal reconstructions.

In parallel, several proprietary software solutions, sometimes directly associated with mobile laser scanner instruments, have been developed to share simplified point cloud visualizations on the web, allowing users to navigate scenes and perform basic measurements (Bai et al., 2025). Other proprietary platforms, such as Pointorama (Fuchs, J., 2025), further enable semi-automatic segmentation of point clouds within a web environment.

Within this framework, the present study proposes an open-source strategy for the rapid extraction of surfaces from laser scanner point clouds and their publication within a web-based visualization environment.

The workflow was designed to operate on datasets characterized by different levels of geometric complexity and to generate a coherent chain from point cloud preprocessing to segmentation, polygonal modeling, mesh construction, and browser-based visualization. The entire process works in a Python environment, in open-source Anaconda platform, then the

browser visualization model is hosted in Apache webserver and employs Three.js JavaScript libraries.
The possibility of automatically publishing segmented polygonal surfaces together with their source point clouds can support professionals in the recognition of the main geometries of surveyed environments and can facilitate subsequent semantic integration processes.
In this sense, the proposed workflow aims to represent a practical contribution toward lightweight and scalable geospatial Digital Twin development based entirely on open-source technologies.
The next paragraph, Materials and methods, illustrates the different steps adopted in the workflow (from the design of the complexity levels to the web visualization structure), then the Discussion paragraph illustrates the different settings adopted in each complexity level, and, finally, the Conclusion paragraph illustrates the scientific gap filled in this experimentation and possible improvements in future scenarios.

2. Materials and methods

2.1 Study design and levels of geometric complexity

The research focused on the development of an automated workflow for surface extraction and web visualization based on laser scanner datasets acquired in environmental contexts with different morphological and architectural characteristics.
In order to evaluate the adaptability of the proposed methodology, three levels of geometric complexity were defined. These levels correspond to progressively more articulated scenarios and guide both the segmentation strategy and the selection of processing parameters (Figure 1).

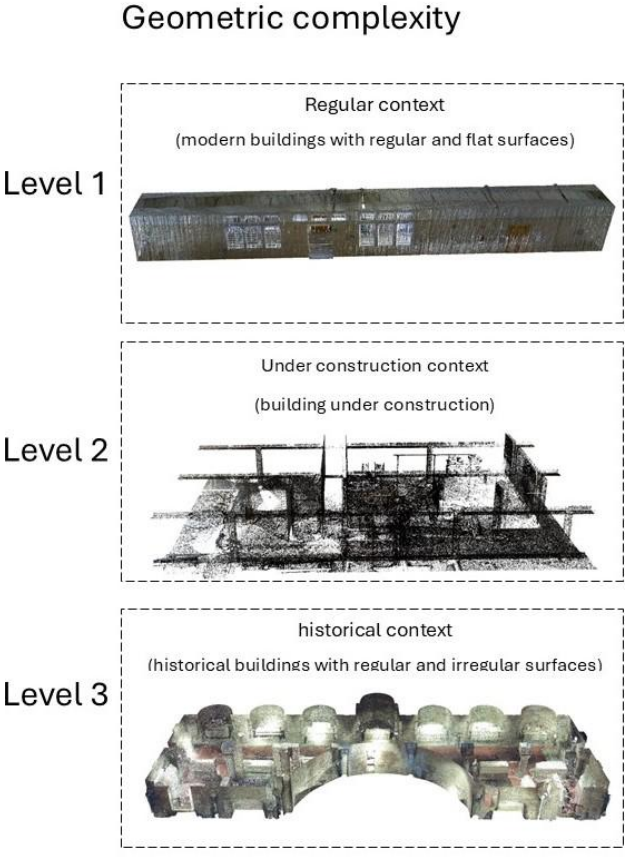


Figure 1. The three levels of geometric complexity.

The first level corresponds to a regular context, represented by a typical modern building mainly characterized by planar and orthogonal surfaces. This level was obtained from a dataset acquired using a mobile laser scanner inside the Department of Engineering building at the University of Palermo (Italy). This dataset is particularly suitable for testing the extraction of dominant planes and for verifying the efficiency of iterative planar segmentation in a relatively controlled geometric environment.
The second level corresponds to an under-construction context get from the mobile laser scanner acquisition of the ITC Building at the University of Twente (The Netherlands) . This dataset presents fewer dominant planes and introduces the presence of pillar elements, offering the opportunity to test the system in a construction site environment.
The third level corresponds to a mixed context, represented by a historical building in which regular and irregular architectural elements coexist. This context is represented by the static laser scanner acquisition of the crypt of the Cathedral of Palermo (Italy). This level is especially relevant because it includes planar walls and floors together with more articulated elements such as arches, vaults, or columns, thus requiring a hybrid processing logic capable of distinguishing between regular and non-regular surfaces.
This subdivision into three levels of complexity was used not only as a descriptive classification of the datasets, but also as an operational framework to calibrate preprocessing choices, segmentation logic, and output generation.
The same general workflow was therefore maintained across the three datasets, while specific parameter settings and extraction strategies were adapted according to the dominant geometric behaviour of each level.

2.2 General workflow architecture

The workflow was implemented within a Python-based processing structure developed in the Anaconda open-source environment. The methodological chain integrated two main processes: on the one hand, the automatic segmentation and surface extraction of the point cloud; on the other, the generation of a web visualization environment capable of hosting both subsampled point cloud data and polygonal segmentation outputs (Figure 2).
The processing relied mainly on the Open3D and NumPy libraries, while the online visualization environment was developed through WebGL technologies, with particular reference to the open-source Three.js library. Final publication was supported within an Apache web server environment.
From an operational point of view, the workflow can be described as an ordered sequence of processing steps:

- First, the raw point cloud was imported and subjected to geometric preprocessing operations intended to improve computational efficiency and reduce the effect of local noise.
- Second, the dataset was analyzed according to its level of geometric complexity, allowing the script to activate the most suitable segmentation logic.
- Third, the extracted surfaces were converted into polygonal representations or lightweight meshes to be implemented in a Json structure.
- Fourth, both the segmented models and a decimated version of the original point cloud were exported into web-compatible formats. In particular, the segmented models were implemented in a Json structure, and the point clouds were saved in ply format.
- Fifth, the outputs were finally integrated within a browser-based visualization environment to enable interactive inspection of the segmented scene.

This ordered structure is particularly relevant because the reliability of the final web representation depends directly on the coherence of the previous preprocessing and segmentation phases.

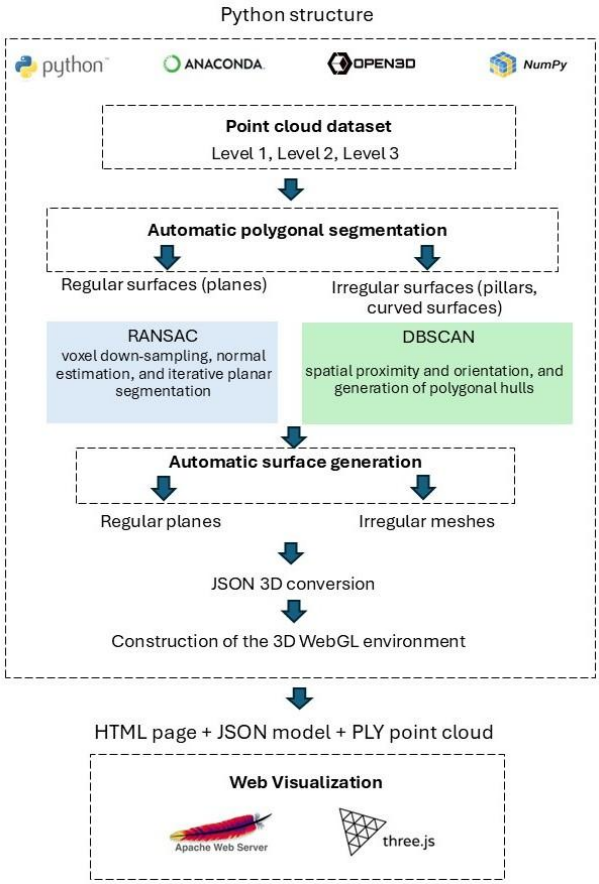


Figure 2. Methodological workflow.

2.3 Point cloud preprocessing

Point cloud preprocessing represented the first essential stage of the Python workflow. Since large laser scanner datasets are often characterized by high density, variable sampling, local outliers, and redundant information, the preprocessing phase was designed to make the subsequent segmentation more stable and computationally efficient.

The first operation consisted of voxel down-sampling, which allowed the reduction of the number of points while preserving the overall spatial structure of the surveyed environment. This step was particularly important to regularize point density and improve the speed of subsequent geometric analyses.

The voxel size can be considered one of the key customization parameters of the workflow, since it directly influences the compromise between geometric detail and computational performance. A smaller voxel preserves more detail but increases processing time and sensitivity to noise, whereas a larger voxel simplifies the dataset and may smooth local geometric variations.

After down-sampling, the workflow performed normal estimation, a fundamental step for identifying the local orientation of surfaces and supporting both planar segmentation and the discrimination of non-planar structures.

The quality of normal estimation depends on the neighbourhood definition adopted in the script, which can be controlled through the search radius or the number of neighbouring points considered. This choice is particularly relevant in mixed and irregular datasets, where unexpected changes in geometry may occur over short distances.

2.4 Point cloud segmentation

The segmentation strategy was adapted to the geometric complexity of the considered dataset. Regular environments were processed through iterative planar segmentation based on RANSAC, while irregular contexts were treated through DBSCAN clustering and polygonal approximation of irregular surfaces. On the basis of this principle, the script adopted three different strategies dividing the overall point cloud environment in three main extraction families: plane extraction, pillar extraction, and curved surface extraction (Figure 3).

The script adopts an order of extraction that starts from the plane extraction, continues with the pillar extraction and ends with the curved surface extraction. At the end of each step the process excludes the part of the point cloud involved in the previous one, in order to avoid overlapping clusters between each extraction.

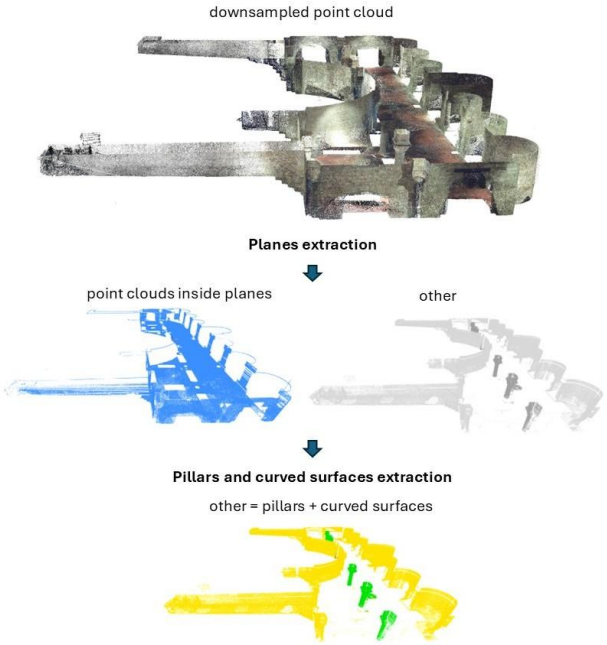


Figure 3. The segmentation process.

The first step considers the iterative planar segmentation. After down-sampling and normal estimation, the cloud was processed through a RANSAC-based search for dominant planes. At each iteration, the algorithm identified the plane best fitting the current point subset, extracted the inliers satisfying the geometric tolerance, and removed them from the cloud before repeating the process on the residual points. This iterative logic allowed the progressive recognition of the main planar elements of the scene, such as walls, floors, ceilings, and other regular surfaces.

The main planar elements were extracted through an iterative RANSAC-based segmentation procedure. At each iteration, the dominant plane was identified according to a user-defined distance threshold and accepted only if supported by a minimum number of inlier points. The process was repeated on the residual points until either a minimum residual size or a maximum number of planes was reached.

The extracted planes were then classified according to the orientation of their normal vectors into horizontal and vertical surfaces. Each accepted plane was simplified into an axis-aligned rectangular patch derived from its spatial extent and further filtered according to minimum area and dimensional

thresholds to retain only the principal wall-, floor-, and ceiling-like components of the scene.
Plane extraction was controlled by the parameters governing RANSAC tolerance (dist_thresh), minimum planar support (min_points), maximum number of extracted planes (max_planes), and the robustness of model estimation (ransac_n, iters).
The resulting planar segments were regularized into simplified orthogonal rectangles and filtered according to minimum area, height, and width constraints, thus enabling the identification of the main structural planar elements of the surveyed environment.
Pillar-like elements were extracted from the residual point cloud after the removal of dominant planar surfaces. The segmentation was performed through DBSCAN clustering applied to the XY coordinates of the residual points, in order to identify compact horizontal point aggregations. Each cluster was then filtered according to its vertical span and horizontal footprint.
Only clusters characterized by sufficient vertical development, minimum planar support, and limited lateral extent were retained as pillar candidates. This strategy allowed the detection of vertically developed compact architectural supports without imposing an idealized cylindrical model.
The segmentation of pillar elements was based on density-driven clustering in the horizontal plane. DBSCAN parameters controlled the neighborhood radius (eps_xy) and minimum local density (min_samples), while subsequent geometric filters evaluated the vertical span of each cluster (min_height), its minimum horizontal footprint, and its maximum lateral extent (max_xy_extent). In this way, pillar candidates were defined as residual point subsets exhibiting planimetric compactness and vertical continuity.
After the extraction of dominant planes and pillar-like elements, the remaining unclassified portion of the point cloud was processed as a residual dataset. In this final segmentation phase, no predefined geometric primitive was imposed.
Instead, the residual cloud was partitioned into spatially coherent 3D clusters through DBSCAN, using Euclidean proximity as the main grouping criterion. This approach enabled the isolation of irregular residual components that could not be reliably represented through planar or pillar-specific extraction rules, that could be considered for the curved surface extraction.
Residual segmentation was performed on the subset of points not assigned to the previously detected planar and pillar-like structures.
DBSCAN clustering in 3D space was used to identify dense and spatially coherent point groups to be considered part of curved surfaces (db_eps, db_min_samples), while a minimum cardinality threshold (min_cluster_points) was applied to discard minor fragments and sparse noise. This phase therefore represented a non-semantic partitioning of the remaining cloud into residual irregular components.
The main point cloud segmentation parameters adopted in the processing are shown in Table 1.

Point cloud Segmentation parameters	
Planes	dist_thresh
	min_points
	max_planes
	ransac_n
	iters
Pillars	eps_xy
	min_samples
	min_height
	max_xy_extent
Curved surfaces	db_eps
	db_min_samples
	min_cluster_points

Table 1. The parameters of segmentation adopted in each step.

2.5 Polygonal and meshes construction

Once the relevant subsets had been extracted, the workflow proceeded with the generation of simplified polygonal models and lightweight meshes. Irregular clusters were converted into polygonal hulls approximating the corresponding surfaces, while all segmented polygonal models were then converted into JSON format for web integration.
After segmentation, each extracted surface subset was converted into a lightweight geometric representation. Dominant planar elements were transformed into polygonal patches derived from their spatial extent, while non-planar clusters were approximated through simplified hull-based surface models. This conversion stage aimed to preserve the interpretability of the recognized surfaces while generating compact outputs suitable for web transmission and interactive rendering.
For the planar components, polygonization was based on the direct regularization of the point subsets extracted through iterative RANSAC segmentation. Plane detection was controlled by the parameters dist_thresh, defining the maximum orthogonal distance tolerated between a point and the candidate plane, min_points, defining the minimum number of inliers required to accept a plane, max_planes, limiting the number of dominant planes extracted from the cloud, ransac_n, defining the number of points used for each candidate model estimation, and iters, corresponding to the number of RANSAC iterations. Once extracted, the planes were converted into simplified polygonal patches.
In this phase, only planes whose normal vectors satisfied orientation thresholds were retained: nz_horizontal_thr was used to identify nearly horizontal surfaces, while nz_vertical_thr was used to isolate nearly vertical ones. Each accepted segment was then transformed into an axis-aligned quadrilateral derived from its spatial extent. For horizontal planes, the polygon was defined by the minimum and maximum X and Y coordinates and by the median elevation of the inlier set; for vertical planes, the rectangle was snapped to the dominant cartesian direction and generated from the median coordinate on the fixed axis and the min-max extents on the remaining axes. A further filtering stage discarded polygons with an area below min_area, while vertical rectangles were additionally constrained by minimum height and width thresholds in order to preserve only the principal wall-like elements.
The final result was therefore not a free-form boundary reconstruction, but a controlled generation of simplified orthogonal planar polygons designed to maximize interpretability and geometric compactness.
For the pillar-like components, reconstruction started from the residual cloud after plane removal and was preceded by a dedicated segmentation step based on DBSCAN clustering in the XY plane. The grouping of candidate pillar points was controlled by eps_xy, defining the neighborhood radius in plan, and min_samples, defining the minimum local density required to form a cluster. The resulting clusters were then filtered according to their geometric characteristics; only subsets with vertical extent greater than min_height, horizontal footprint area above a minimum threshold, and maximum lateral extent lower than max_xy_extent were retained as pillar candidates. Once segmented, each cluster was converted into a triangulated surface through a local mesh reconstruction process. Before meshing, normals were estimated using a hybrid KD-tree search and then consistently oriented through tangent-plane

propagation where possible. Mesh generation was performed through ball pivoting in order to adapt the triangulation to local variations in point spacing and object thickness. The generated surface was then cleaned through topological filters removing duplicated vertices, degenerate triangles, non-manifold edges, and unreferenced vertices. Only meshes with a minimum number of triangles were retained. From a methodological point of view, pillar polygonization can therefore be described as a cluster-based local mesh reconstruction, in which the segmented point subset is transformed into a lightweight triangulated envelope preserving the vertical morphology of the support without imposing an idealized parametric cylindrical model.

For the residual irregular and curved components, the reconstruction phase followed a more general strategy. After the extraction of planes and pillar-like elements, the remaining cloud was treated as an unclassified residual dataset and segmented through DBSCAN directly in 3D space. In this phase, clustering was controlled by db_eps, defining the Euclidean neighborhood radius, and db_min_samples, specifying the density threshold required for cluster formation. Clusters with fewer than min_cluster_points were discarded, so that only spatially coherent residual subsets were preserved for reconstruction. Each accepted cluster was then converted into a lightweight triangulated surface. Before meshing, normals were estimated maintaining the scale of normal estimation adaptive to the sampling density defined by voxel. Normal orientation consistency was then improved through tangent-plane propagation. After reconstruction, the mesh was topologically cleaned and retained only if the number of generated triangles exceeded min_triangles. In this sense, the polygonization of curved or irregular components did not correspond to the extraction of an analytical primitive, but to the generation of a simplified surface envelope derived directly from the clustered residual geometry.

Overall, the polygonal reconstruction stage was differentiated according to the geometric nature of the extracted subsets and to the parameters controlling each process. Planes were transformed into orthogonal rectangular polygons governed by orientation and minimum extent thresholds; pillars were reconstructed as compact triangulated meshes derived from vertically developed residual clusters, and irregular residuals were approximated through adaptive mesh generation based on either alpha shapes or multi-scale ball pivoting. This parameter-driven strategy enabled the workflow to maintain a balance between geometric simplification, morphological fidelity, and lightweight exportability for web-based visualization.

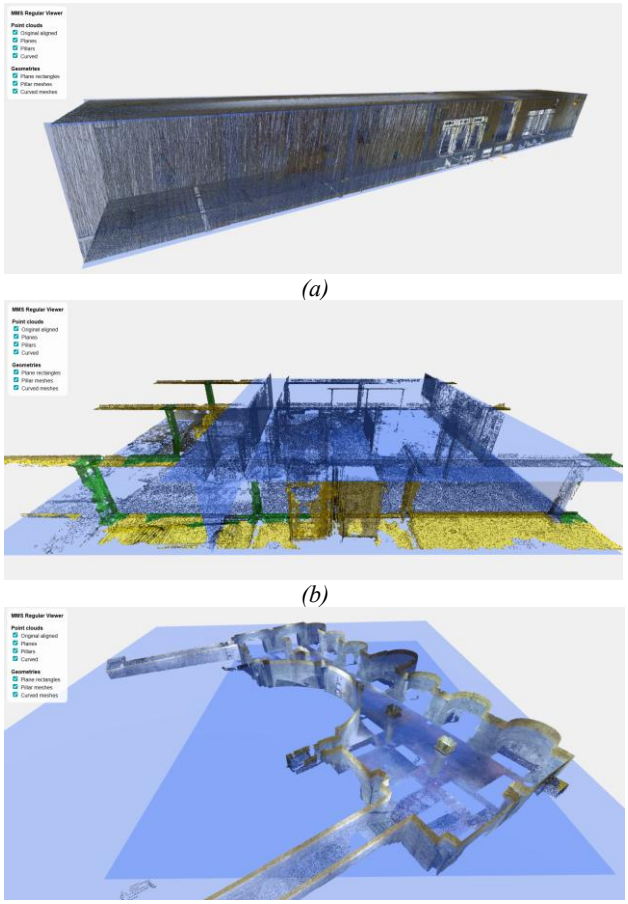
2.6 Web visualization environment

The last methodological stage concerned the creation of the web environment. The interface combined three main outputs of the Python process: the HTML page, the JSON segmentation model, and the PLY point clouds, all integrated within an Apache web server environment. The visualization exploited WebGL through Three.js, allowing the simultaneous display of the source point cloud and the extracted polygonal surfaces. This part is worth emphasizing because it is not merely a visualization add-on, but the final destination of the whole pipeline.

The interface allows users to remotely navigate in a 3D web browsing experience that integrates for each level the visualization of:

- The downsampled point cloud (PLY)
- The planes segmentation (PLY)
- The pillars segmentation (PLY)
- The irregular surfaces segmentation (PLY)
- The main planes extracted (.JSON)
- The pillar meshes (JSON)
- The curved meshes (JSON)

The visualization adopts a menu bar that allows users to switch between each element visualized, combining the visualization of point cloud segmentation and segmented surface extraction (Figures 4).



(a) *(b)* *(c)*

Figure 4. The web visualization levels: (*a*) Level 1, (*b*) Level 2 and (*c*) Level 3.

3. Discussion

The results demonstrate that the proposed workflow can support the automatic extraction and web-based publication of significant geometric surfaces from laser scanner point clouds across different environmental conditions. One of its main strengths lies in the adoption of a single open-source processing chain that can nevertheless be adapted to distinct geometric scenarios. Rather than relying on a rigid and universal segmentation procedure, the method organizes the processing according to the complexity of the surveyed environment, distinguishing among regular, mixed, and irregular cases. This makes the approach particularly promising for practical applications in architecture, cultural heritage, underground spaces, and geospatial Digital Twin development.

From a methodological perspective, the distinction between planar extraction and irregular clustering proved essential. In regular contexts, RANSAC-based plane detection can effectively extract a limited number of meaningful surfaces from the point cloud (Figure 7). However, in built heritage environments, this logic alone is not sufficient, because a relevant part of the geometry is represented by curved, eroded,

or non-standard structures. The integration of density-based clustering and hull-based approximation therefore extends the applicability of the workflow beyond ideal architectural spaces and improves its robustness in real-world scenarios (Figure 8).

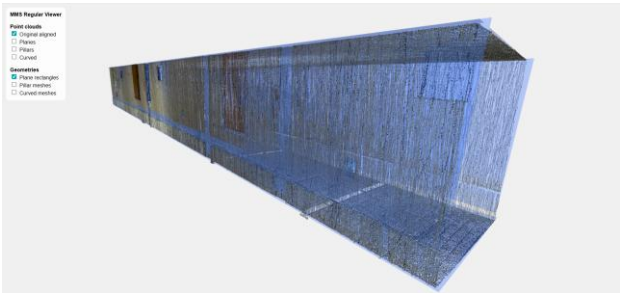


Figure 7. Level 1 web visualization of the regular environment with the Ransac-based the main surfaces (in blue) extracts from the point cloud.

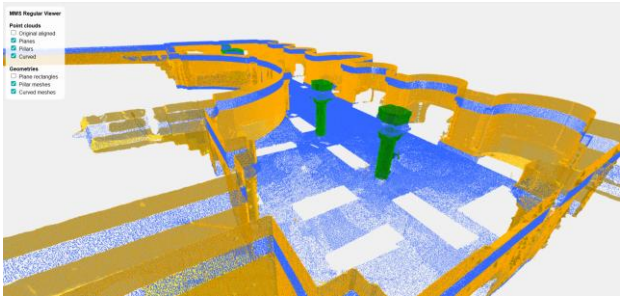


Figure 8. Level 3 web visualization of the built heritage environments with irregular surfaces wherethe point cloud was divided in planes (blue), pillars (green) and irregular surfaces (yellow).

Another important aspect concerns the relationship between geometric simplification and web usability. Full-resolution point clouds remain difficult to manage efficiently in browser environments, especially when interactive rendering must be preserved on standard hardware. By extracting and publishing lightweight segmented surfaces together with a reduced point cloud representation, the workflow provides an effective compromise between geometric richness and visualization performance. In this sense, the web interface is not intended to replace the complete metric dataset, but to provide an accessible and semantically more readable layer for online exploration and communication.

At the same time, some limitations should be acknowledged. The accuracy and stability of segmentation results depend on the quality of the original acquisition, the level of noise, the local density of points, and the parameterization adopted in the script. The choice of voxel size, plane tolerance, clustering thresholds, and mesh generation criteria may significantly affect the final output. Moreover, mixed contexts remain particularly challenging because they require a balanced transition between regular and irregular extraction logics. For this reason, the workflow should be interpreted as a scalable and customizable framework rather than as a fully universal solution.

A further point of interest lies in the semantic potential of the generated outputs. In fact, the automatic publication of segmented polygonal surfaces can facilitate semantic integration processes. In future developments, the extracted planar, pillar-like, and curved components could be associated with semantic labels and connected to databases or web services, thus contributing more directly to multi-source geospatial Digital Twin systems and HBIM-oriented environments.

4. Conclusions

This study presented an open-source workflow for the automatic extraction of surfaces from large laser scanner point clouds and their integration within a web-based visualization environment. Developed in Python and structured around a level-based interpretation of geometric complexity, the workflow combines point cloud preprocessing, adaptive segmentation, polygonal or mesh-based surface generation, and lightweight web publication. The methodology was conceived to operate across three different classes of datasets, ranging from regular modern buildings to mixed historical contexts and highly irregular hypogeal environments.

The obtained results show that the proposed approach can provide an effective balance between automated geometric recognition and browser-oriented simplification. In regular scenes, the workflow supports the reliable extraction of dominant planes; in mixed scenes, it improves the distinction between planar and articulated architectural elements; and in irregular environments, it enables the approximation and online visualization of non-standard surfaces through density-based clustering and simplified polygonal reconstruction. The final web environment, based on open-source WebGL technologies, demonstrates the practical feasibility of publishing both segmented models and point cloud data in an accessible and scalable way.

Overall, the workflow represents a promising contribution for low-cost 3D dissemination, online interpretation of surveyed spaces, and future semantic enrichment of geospatial web applications. Future developments may focus on the more robust recognition of architectural components such as pillars and beams, on the refinement of curved surface modeling, and on the integration of semantic descriptors and database connections, further strengthening the value of the approach for Digital Twin and cultural heritage applications.

References

Alsadik B., Karam, S., 2021: The Simultaneous Localization and Mapping (SLAM)-An Overview. *JASTT*, vol. 2, no. 02, pp. 147–158. doi: 10.38094/jastt204117.

Aminiti, P., Bonora, V., Corongiu, M., Mugnai, F., Parisi, E.I., Tucci, G., 2020: Geomatics techniques for the 3D survey of the Arno River to support hydraulic studies. *IOP Conf. Ser.: Mater. Sci. Eng.,* Volume 949. DOI 10.1088/1757-899X/949/1/012104

Aricò, M., La Guardia, M., Lo Brutto, M. 2023: 3D Data Integration for Web Fruition of Underground Archaeological Sites: A Web Navigation System for the Hypogeum of Crispia salvia (Marsala, Italy). *Heritage*, 6, 5899-5918. https://doi.org/10.3390/heritage6080310

Aricò, M., Dardanelli, G., La Guardia, M., Lo Brutto, M. 2024: Three-Dimensional Documentation and Virtual Web Navigation System for the Indoor and Outdoor Exploration of a Complex Cultural Heritage Site. *Electronics*, 13, 2833. https://doi.org/10.3390/electronics13142833

Bai, X., Zhang, T., Shui, R., Kang, X., 2025: Real-time 3D scene reconstruction of ancient opera stage architectural heritage using a handheld SLAM laser scanner: a case study of the Lixel L2 Pro. *Proc. SPIE 13793, Fourth International Conference on Machine Vision, Automatic Identification, and Detection (MVAID 2025)*, 137931I. https://doi.org/10.1117/12.3077373

Fuchs, J., 2025: Scan to BIM: Capturing and Modeling Existing Buildings. Implementation and Evaluation of Methods for Automatically Converting Point Clouds into BIM Models. Master’s thesis, Hochschule für angewandte Wissenschaften München, Munich, Germany.

Gaspari, F., Barbieri, F., Fascia, R., Ioli, F., Pinto, L., 2024: An Open-Source Web Platform for 3D Documentation and Storytelling of Hidden Cultural Heritage. *Heritage*, *7*(2), 517-536, https://doi.org/10.3390/heritage7020025.

Gollob, C., Ritter, T., Nothdurft, A., 2020: Forest Inventory with Long Range and High-Speed Personal Laser Scanning (PLS) and Simultaneous Localization and Mapping (SLAM) Technology. *Remote Sens*., 12(9), 1509. https://doi.org/10.3390/rs12091509.

Haznedar, B., Bayraktar, R., Ozturk, A.E., Arayci, Y., 2023: Implementing PointNet for point cloud segmentation in the heritage context. *Herit Sci* 11, 2. https://doi.org/10.1186/s40494-022-00844-w.

Huang, Z.J., Mei, G., Zhang, J., Abbas, R., Landrieu, L., 2024: Surface Reconstruction From Point Clouds: A Survey and Benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12), pp. 9727-9748. DOI: 10.1109/TPAMI.2024.3429209.

La Guardia, M., Koeva, M., D’Ippolito, F., Karam, S., 2022: 3d Data Integration For Web Based Open Source Webgl Interactive Visualisation. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., XLVIII-4/W4-2022, 89–94, https://doi.org/10.5194/isprs-archives-XLVIII-4-W4-2022-89-2022.

La Guardia, M., Masiero, A., Bonora, V., Alessandrini, A., 2025: Open Source Deep Learning Solutions for the Classification of MMS Urban 3D Data. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci*., XLVIII-G-2025, 839–844. https://doi.org/10.5194/isprs-archives-XLVIII-G-2025-839-2025.

Masiero, A., Fissore, F., Guarnieri, A., Vettore, A., Coppa, U., 2020: Development and Initial Assessment of a Low Cost Mobile Mapping System. *R3 in Geomatics: Research, Results and Review. R3GEO 2019. Communications in Computer and Information Science*, vol 1246. Springer, Cham. https://doi.org/10.1007/978-3-030-62800-0_10

Matellon, A., Maset, E., Beinat, A., Visintini, D., 2026: How the Choice of SLAM Algorithm Impacts Point Cloud Classification: An Experimental Evaluation. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci*., XLVIII-2/W12-2026, 295–302, https://doi.org/10.5194/isprs-archives-XLVIII-2-W12-2026-295-2026.

Sulzer, R., Marlet, R., Vallet, B., Landrieu, L., 2025: A Survey and Benchmark of Automatic Surface Reconstruction From Point Clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(3), pp. 2000-2019. DOI: 10.1109/TPAMI.2024.3510932.