SCIentificRESearch and Information Technology
RicercaScientifica e Tecnologiedell'Informazione
Vol 4, Issue 2 (2014), 17-28
e-ISSN 2239-4303, DOI 10.2423/i22394303v4n2p17

ONLINE ACCESS AND SHARING OF REALITY-BASED 3D MODELS

Simone Minto, Fabio Remondino**

*3D Optical Metrology unit, Bruno Kessler Foundation, Trento, Italy

Abstract

The article presents an investigation on techniques and solutions for publishing reality-based 3D models online. The process starts from a dense point cloud and leads to a reduced textured 3D model accessible on the web with a browser. The work is divided into four phases: (i) generation of a polygon mesh model, (ii) 3D model segmentation, (iii) simplification of the polygonal model (geometry and texture) in different levels of detail (LoD) and (iv) publication on the web. The entire workflow is based on Open Source software. Comparisons with commercial solutions are also reported.

Keywords

Point cloud, polygonal mesh, texture, segmentation, semplification, web access, web publication, PDF3D, WebGL

1. *Introduction*

Modern automated techniques of Image-Based Modeling (IBM) and more traditional acquisitions with active / ranging instruments (Range-Based Modeling – RBM) have led to a considerable use of 3D digital data (point clouds or polygonal models) also in the context of Cultural Heritage [Remondino and Campana, 2014; Remondino et al., 2013; Guidi et al., 2013; Gallo et al., 2013; Menna et al., 2012; Stanco et al., 2011]. This opens up new possibilities and interesting approaches for analyses, preservation, restoration and promotion. Moreover the latest developments in 3D data optimization, web publication, 3D models sharing and 3D printing is opening new avenues for remote access, digital hypothetic reconstruction, AR applications as well as physical replicas or heritage contents distribution and dissemination [Scopigno et al., 2014; Potenziani et al., 2014; Jiménez Fernández-Palacios, 2014; Landrieu et al., 2011; Manferdini and Remondino, 2011; Manferdini et al., 2008].

This article presents the development of a methodology [Fig. 1] for publishing and sharing 3D models on the web. The methodology involves mainly the use of open software and free viewers. The work is not addressing the 3D acquisition phase, but it starts from available point clouds derived in some case studies conducted in a collaborative research project (TAPENADE - http://www.tapenade.gamsau.archi.fr; Pierrot-Deseilligny et al., 2011) between IGN MATIS Laboratory (Paris, France), CNRS MAP-Gamsau Laboratory (Marseille, France) and 3DOM-FBK (Trento, Italy). The dense point clouds were generated using image datasets and the open source suite MicMac (http://micmac.ign.fr/; Pierrot-Deseilligny, 2014).

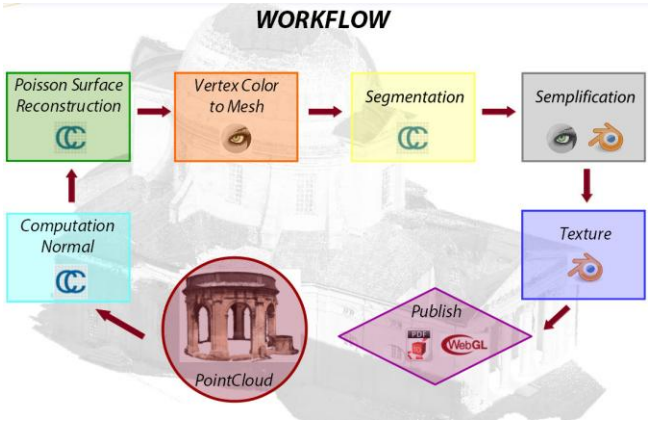


Fig. 1: Adopted workflow, mainly based on open solutions.

2. *Objectives*

The aims of the work are multi-folds:

- investigate the geometric modeling of photogrammetric dense point clouds;
- consider the solutions for publishing optimized and texturized 3D models on the web;
- overcome the difficulties related to large sizes of 3D models with the bandwidth limits of the actual web connections.

The work will report an in-house pipeline, based mainly on open source solutions, although commercial tools are also presented.

3. Point clouds and meshes

The workflow [Fig. 1] starts from the available dense and unstructured point clouds (with RGB information) produced following specific guidelines [Fig. 2; Nony et al., 2012]. The pipeline consists of different steps for creating a structured, segmented and optimized geometric model for web publication and access. As the considered point clouds are generated with a photogrammetric approach, the normal are initially computed before generating the polygonal surface and post-process it.

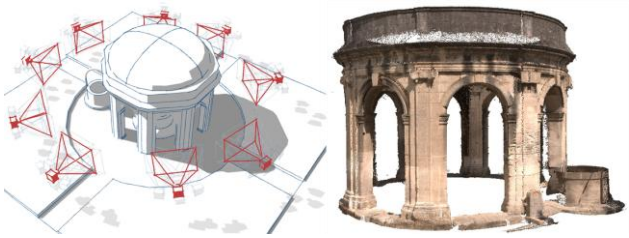


Fig. 2: Image acquisition protocol (left) and produced dense 3D point cloud for the Fountain of Saint-Jean (right).

3.1.1 Normal computation

A normal is a vector perpendicular to a point lying on a surface (i.e. the tangent plane of the point). In computer graphics normal are normally used to increase the realism of rendering (as it "reflects light") and their direction indicates what is considered the “inside” and the “outside” of a surface. Depending on the employed geometric algorithm, normal are often required for the generation of the polygon mesh. Normal can be estimated analyzing the neighboring points, setting a value of radius around the considered point [Fig. 3].

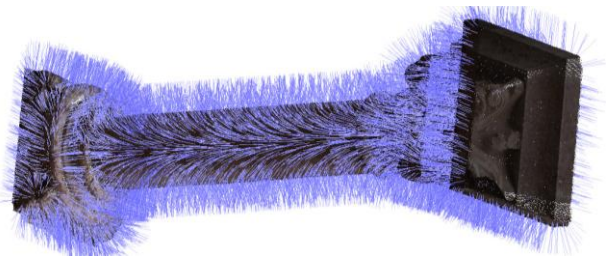


Fig.3: Computed normal on a dense point cloud of a column (software: CloudCompare 2.5.1; command: Compute normal).

3.1.2 Mesh generation with the Poisson algorithm

A polygon mesh is essentially a collection of vertices, edges and faces that define the shape of a three-dimensional object. A vertex is the representation of a position in space. An edge is the entity that connects two vertices. A face is a set of points in the space enclosed between edge and vertices. The set of these faces can determine polygons or much more complex structures. The faces usually consist of triangles, although quadrangular shapes are also used. In a mesh, in order to be defined as such, one edge is shared by at most two faces and is between two vertices. While a vertex is shared by at least two edges. A mesh is defined closed (or watertight), if there are no holes on its surface, otherwise it is normally named open. In these cases, all the elements (edges, vertices, etc.) that they find themselves on the edge are defined boundary.

The mesh can be obtained in various ways and with various techniques [Remondino, 2003; Berger et al., 2014] starting from:

1. points (triangulation algorithms and tetrahedralization e.g. Delauny, Poisson, etc.);
2. surfaces (tessellation algorithms, in particular for rendering curved surfaces);
3. data volumes (volumetric data, isosurfaces, etc.).

Nowadays a very well known and used algorithm for mesh generation is the Poisson surface reconstruction method [Kazhdan et al., 2006]. It requires as input a point cloud with normal information and follows some connected steps to produce a polygonal surface. Typically, for every point of the input dataset, it makes an average of the neighborhood normal in order to reduce the noise. Starting from the set of all normal, the algorithm derives an initial shape solving what is seen as a Poisson problem, hence the name. By its nature, however, the algorithm performs better for watertight surfaces. If the initial data is not such a surface, the algorithm tries to find a way to close it, often adding useless areas and forcing a manual post-processing and cleaning operation. The levels of resolution of the algorithm depends on an single parameter (octree) defining a tree data structure where each internal node has a given number of children (default 8). The values of octree used and compared are 8, 9, 10, 12, 14 [Fig. 4].

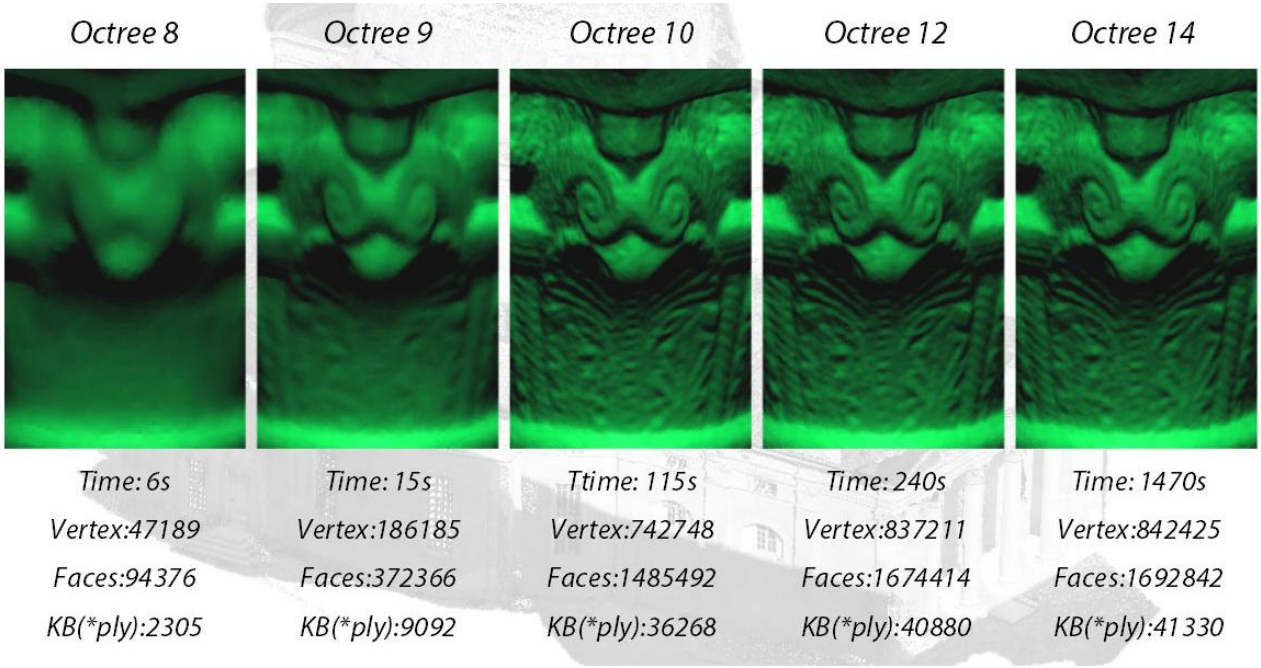


Fig. 4: Comparison of different mesh resolutions generated by the Poisson algorithm with different octree sizes (software: CloudCompare 2.5.1; command: Poisson Reconstruction).

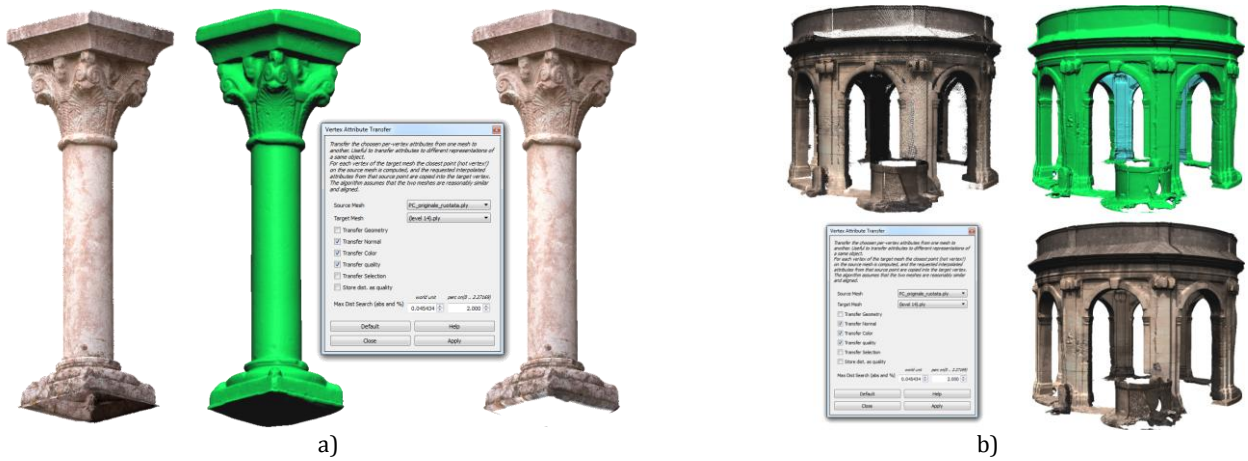
Already with a value of octree equal to 10 the quality of the polygonal model is satisfactory.A further increase of octree does not bring, usually, any improvement in the geometry (the number of faces increases slightly), but only an increase in processing time. The octree size should reflect the average point distance in the cloud.

3.1.3 Transfer of color information

The Poisson's algorithm is not able to handle color information and the generated mesh is colorless. To obviate this drawback, color information needs to be associated to the geometry information using a vertex to vertex interpolation from the original point cloud [Fig. 5]. This transfer operation could decrease the radiometric quality of the model (with respect to a color projection directly from an image to the mesh polygons) if the face dimensions are too large.

3.2 3D model segmentation

3D geometries need to be segmented in order to allow easy consultation and access, insertion of external information, identification of different



a) b)

Fig. 5: Two examples of the color information transfer from the original point cloud to the polygonal model (software: MeshLab 1.3.2; command: Vertex Attribute Transfer).

construction period or material, etc.
Segmentation means to extrac geometric information from a set of data and the generation of uniform entities. The segmentation can be performed on the unstructured point cloud [Oehler et al., 2011] or on the structured geometric model. In the former case, each of segmented entity will be modeled separately and the final 3D model of the object will be composed from all of the various portions separately modeled. Segmentations of polygonal models lead to many troubles in the entity border areas as it is very complicate to correctly and automatically separate faces.

3.2.1 Automatic segmentation

There are many researchers investigating automatic segmentation of 3D geometries. Most of the work is done on LiDAR data (e.g. DEM) acquired from aerial platforms. In terrestrial applications data are much more complex but some algorithms and segmentation techniques developed for DEM segmentation can be adapted and adopted. Among the possible methodologies to automatically segment 3D data, we can report:

a) Segmentation by identifying the principal planes of the object

In order to segment and model 3D data, it is firstly necessary to identify its principal planes. In case of a building, the main planes can be represented by the single façades or by portions of them in the case of more complex structres. A typical interactive but reliable procedure to detect planes is based on the following schedules:

- a reference point in the analyzed point cloud is considered to start the process.
- a search radius around the starting point is selected and all the points belonging to this area are used to determine plane passing through the selected points [Fig. 6].
- iteratively and using user-defined parameters all the main plains are sought.

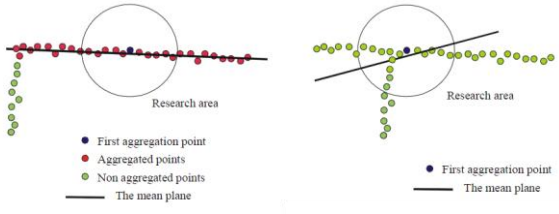


Fig. 6: Determination of a median plane of a point cloud.

b) Segmentation techniques using region growing and principal component analysis

These approaches try to group sets of individual entities that have certain common properties. The algorithms seek the solution using clustering techniques that originate from the techniques of image analysis.

c) RANSAC

RANSAC is an iterative method for the estimation of parameters of a mathematical model from a set of input data, possibly containing a large percentage of outliers. It 'a non-deterministic algorithm, that produces a correct result only with a given probability, which increases with the increase of the iterations. These methods are fully automated as they generate candidate solutions using a minimum number of observations (input data) needed to estimate the parameters of the model. In our experiences this method was not satisfactory, thus leading us to prefer a manual segmentation solution more suited to a subdivision according to the architectural vocabulary [Fig. 7].

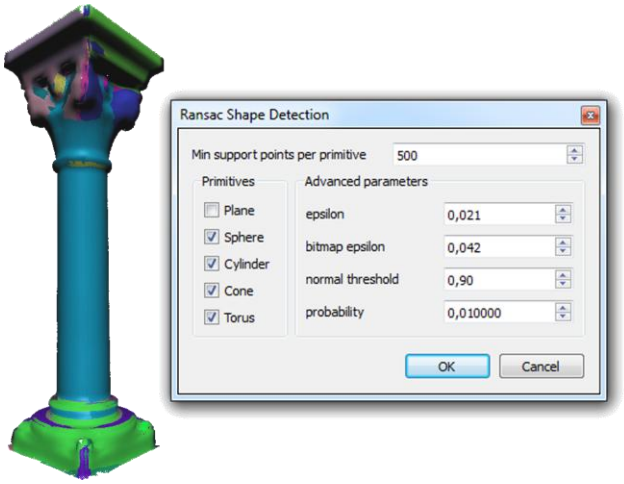


Fig. 7: RANSAC algorithm applied to the point cloud of a column (software: CloudCompare 2.5.1; command: Ransac Shape Detection). Due to the complex shape of the object, unsatisfactory segmentation results are achieved.

3.2.1 Manual segmentation

Although time consuming, a manual approach allows to identify the main geometric elements and, if necessary, to follow rules dictated by the architectural language (column, pillar, arc, etc.). This operation can be performed on a point cloud or directly on a mesh model. In our experiments [Fig. 8] we focused directly on polygonal models.

a) b)

Fig. 8: Examples of polygonal model segmentation (software: CloudCompare 2.5.1; command: Segment).

3.3 Simplification of polygonal meshes

The growing number of high resolution textured 3D models, produced to increase the realism of digital scenes, has as its counterpart the difficulty to view these scenes interactively and maybe online. The possible solutions are: (i) increase the performance of hardware platforms, (ii) create multi-resolution models [Di Benedetto et al., 2014] or (ii) use simplification techniques for geometry and texture.

Nowaday the main limitation is the internet's bandwidth limit therefore our choice was directed towards data simplification and optimization for web access, particularly testing two possible solutions: decimation and remeshing.

The mesh simplification describes a class of algorithms that, starting from a mesh as input, deliver another one with less faces, edge and vertex. Most of the simplification techniques proposed in the literature are based on some variations or combinations of two basic basic mechanisms: decimation and union leaders. However, there are also approaches based on sampling and adaptive subdivision. This is a possible classification of these methods:

1. *Iterative decimations* remove vertices or faces of the mesh, re-triangulating the resulting hole within every step. Such algorithms are relatively simple to implement, can be very fast and are excellent for the elimination of redundant elements, such as the coplanar faces.
2. *Union of vertices* (vertex-merging) collapse two or more vertices into a single new one which may be joined to other vertices in a later stage. The merge of the vertices of a triangle is equivalent to the elimination of the triangle, i.e. the reduction of the number of faces. The efficiency of these methods often depends on which scheme is used to decide which vertices must be joined and in which order. A subclass of these methods is based on edge-collapsing.
3. *Adaptive subdivisions* calculate a simple base mesh which is recursively divided to approximate more and more the original surface model. This approaches work better if the base model is simple to be calculated. These methods preserve the topology, which may limit their use in case aggressive simplifications are required. On the other hand they are particularly suited for editing multi-resolution surfaces, since the changes to the levels of detail is automatically propagated up to higher levels.

In our case, we focused on two possible solutions, one with an edge-collapse algorithm and one based on remeshing with recursive resampling.

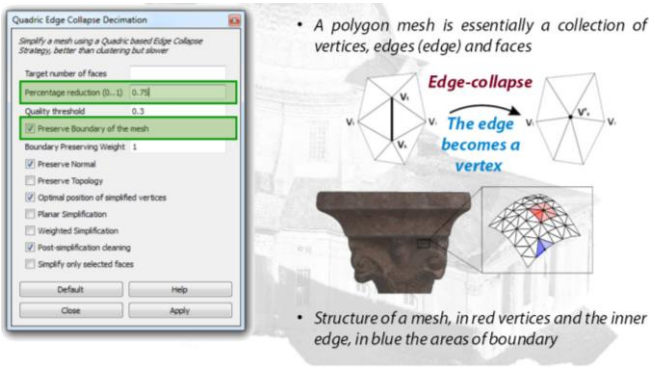


Fig. 9: Principle of the edge-collapse method.

3.3.1 Quadric edge collapse decimation

A selected edge is collapsed into a single vertex and the two faces between the edge are eliminated [Fig. 9]. Various resolutions of simplification can be selected, being careful not to

simplify the mesh boundary, thus avoiding holes between the individual segmentation models. Figure 10 shows an example of mesh segmentation with the edge collapse method and different levels of simplification (75%, 50%, 25%, 10%, 1%).

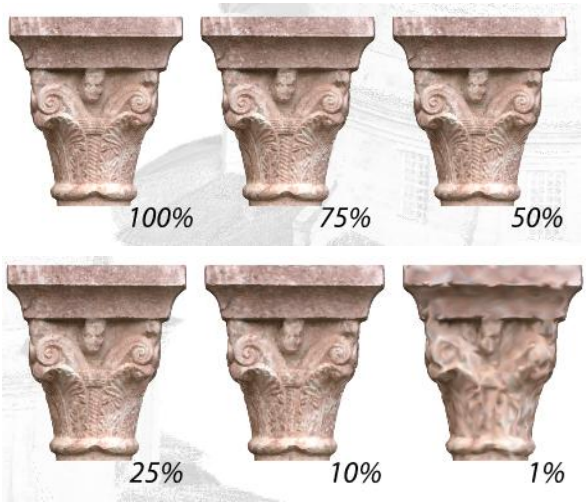


Fig. 10: Comparison between the various levels of decimation (software: MeshLab 1.3.2, comand: Quadric Edge Collapse Decimation).

3.3.2 Remesh - Marching Cubes

The employed method exploits a sampling algorithm, based on a variant of Marching Cubes, which is used normally for the construction of 3D surfaces with constant density. The more the value of octree is increased, the higher is the sampling and the resolution of the model [Fig. 11]. Unfortunately, the process of Remesh looses the color information, so the vertex attribute transfer should be repeated again.

3.3.1 Comparison of simplified meshes

The simplified mesh models obtained with the two aforementioned methods were afterwards compared. A visual comparison shows that the results from the edge-collapse approach is much more "disordered" [Fig. 12].

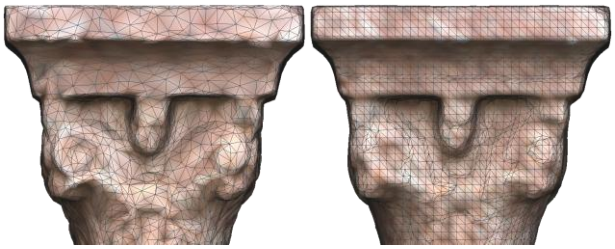


Fig. 12: Visual comparison of the simplification results from the edge collapse (left) and remesh (right) methods.

In addition to this visual comparison, various simplified meshes were geometrically compared to assess the level of simplification [Fig. 13]. Table 1 reports the differences among the two methods for the column 3D model. It is clearly visible how the Remesh approach presents deviations from the original model in the order of 20 times compared to the error obtained by decimation always compared to the default.

Tab. 1: Comparative results of geometrical differences between original and simplified mesh.

Mesh	Faces	Vertices	KB (*.ply)	Avg. dist. [m]	Standard Deviation
Original model	1646080	823045	43403	-	-
Decimate 75%	1235417	617263	32552	0	0,002
Decimate 50%	823000	411504	21701	0	0,005
Decimate 25%	***411482***	***205745***	***10851***	***0.001***	***0.013***
Decimate 10%	164594	82301	4341	0.004	0.035
Decimate 1%	16452	8230	435	0.056	0.241
Remesh 9	521224	994356	37692	-0.010	0.074
Remesh 8	***129652***	***249062***	***9430***	***-0.042***	***0.259***
Remesh 7	32156	62380	2358	-0.146	0.722

a) b)

Fig. 11: Two examples of Remeshsimplification with different values of octree (software: : Blender 2.68, comand: Remesh – Marching Cubes.

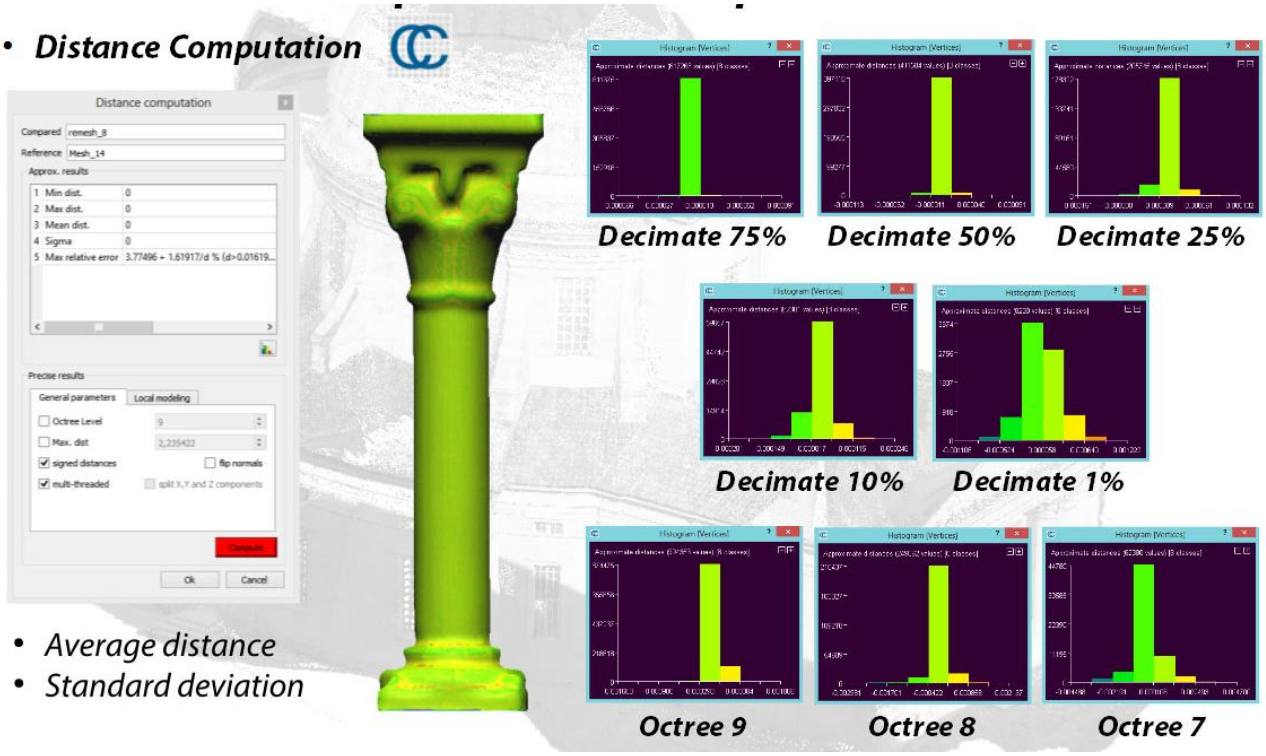


Fig. 13: Geometric comparison between the simplified mesh (software: CloudCompare 2.5.1, comand: Distance Computation)

3.4 Texuture

During the simplification step, the quality of the RGB values assigned to the mesh decresed significantly. Hence the need to transfer again the chromatic qualities of the original model to the simplified geometry. This is a mandatory step in order to keep rendering quality high and preserve good color knowledge, a very useful information for cultural heritage 3D models.

For complex 3D models, the cube, cylindrical or spherical mapping are usually not sufficient. A more accurate projection and texture mapping is normally achieved with the UV mapping method. A UV map describes which part of the texture is attached to which polygon of the model. Each vertex of the polygon is assigned to 2D coordinates that define which part of the image is mapped. These 2D coordinates are called UV (compared to the XYZ coordinates of the 3D system). The process of generating these UV maps is also called "unwrap" (or unwinding), as it is like the mesh is open on a 2D plane [Fig. 14a].

The color information and then mapped onto this unwrap mesh to produce a unique texture map of the polygonal modal with a direct / linear link among geometry and color [Fig. 14b]. This method can preserve good texture quality and even allow to associate low geometries to high color information. The negative aspects of this process are due to the long operational time for the unwrapping operations (several minutes for complex models).

3.5 Publish

3D textured models should be shared and make available to a large variety of users for analyses and studies, through comprehensive professional software or through specific viewers. In particular, non-experts should have easy and clear access to 3D information, possible online or through a web browser. Actual available solutions (languages, libraries, API) to view and share 3D content on the web are:

1. 3DPDF
2. WebGL / HTML5
3. Remote Rendering and Serius Games
4. Pseudo 3D
5. Solutions All in One (NUBES, PointCab, Sketchfab, Memento, 3DHOP, etc,)

In the following sections, some experiences with PDF3D and WebGLare presented.

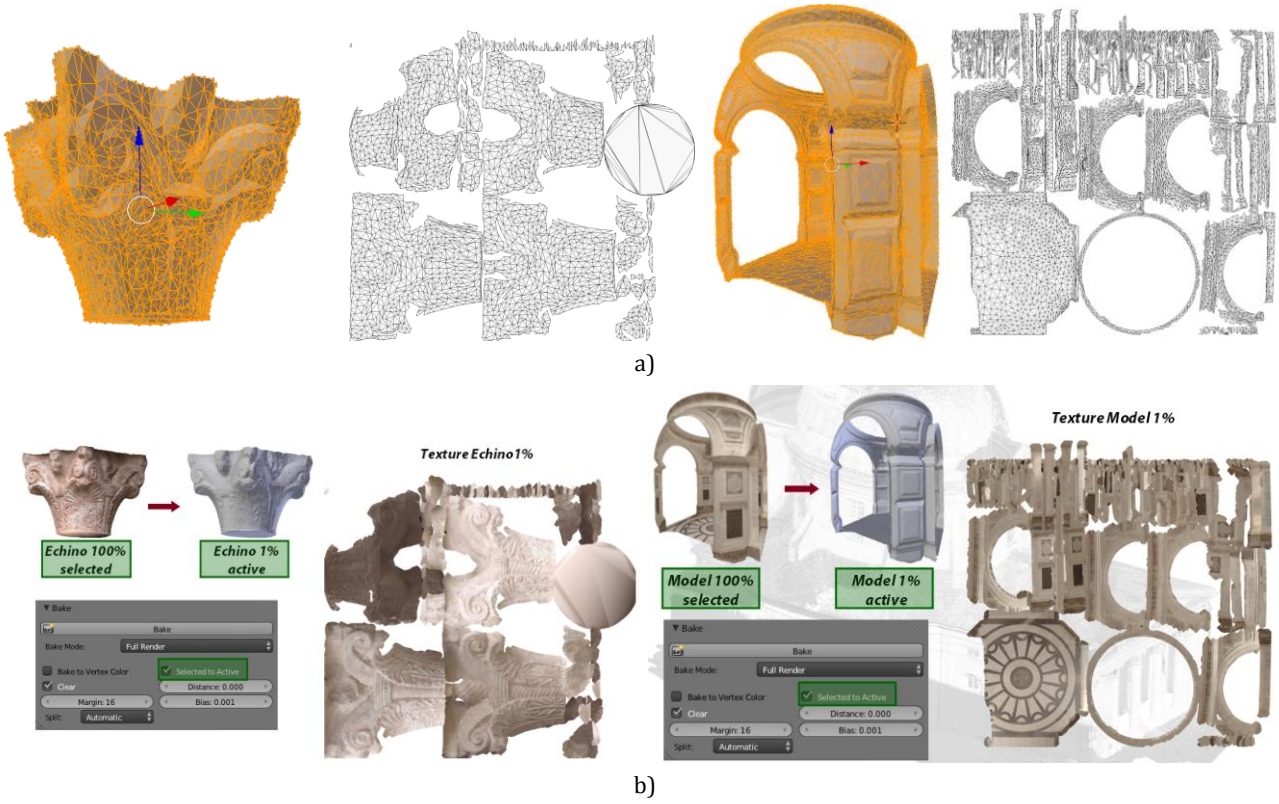


Fig. 14: Two examples of mesh models and their planar unwrapping (a). For the same models the UV map textures are produced (b) (software: Blender 2.68, command: Bake).

3.5.1 3DPDF

The Portable Document Format (PDF) is without doubt one of the most commonly used formats for the exchange and display of electronic documents in all professional contexts. In a PDF file, you can embed a 3D model generated by CAD or modeling programs. To view the objects and interact with them is a sufficient common PDF viewer (though some viewers and browsers have still some problems in displaying it).The 3DPDFcan be a valuable tool for the dissemination of cultural heritage. Indeed, links, textual information and segmented parts (i.e. layesr) can be also imported and various disply mode texture, wireframe, solid, etc.) are available.

On mobile devices and operative systems there are currently no solution PDF3D.

Some modeling tools allow already to export a 3DPDF file. In our experiments, two stand-alone solutions were tested, one open source and proprietary.

a) 3DPDF OpenSource

U3D2PDFis an open tool which allows to automatically convert STL and VRML 3D models into 3DPDF files [Fig. 15]. U3D2PDF requires the installation of MeshLab and MiKTEX (software for typesetting). The resulting file uses the considerable compression of the U3D format and can reduce the file size to 1/10th, although the final product loses the texture information (or vertex color) and the segmentation information (as not supported by the input vrml and stl format). Remain to be exploited all the features of navigation and the ability to take action [Fig.15]..

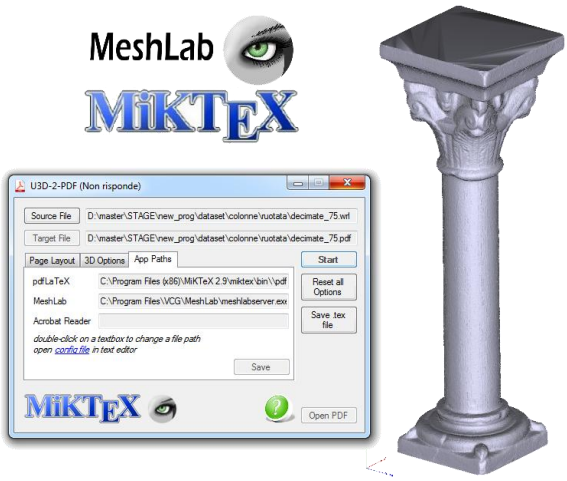


Fig. 15: Open Source pipeline for generating 3DPDF: the resulting model loses information and color segmentation, but takes advantage of the considerable compression of the U3D model (software: U3D2PDF - MeshLab+MiKTEX).

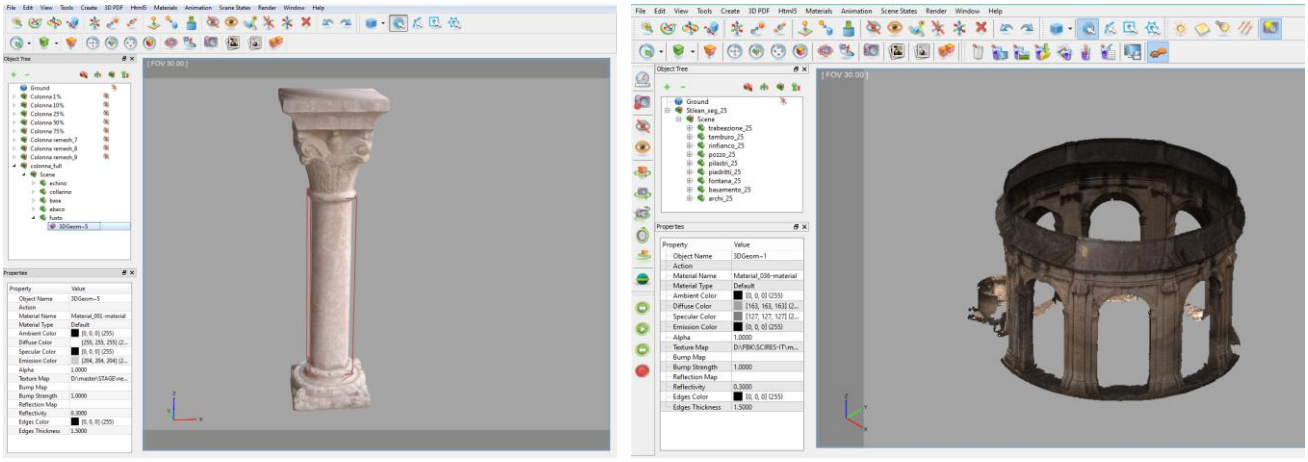


Fig. 16: The GUI of SimLab which allow to maintain the model segmentation, texture as well as to assign links.

b) SimLab Composer

SimLabComposer is a commercial software of SimLabSoft. The software [Fig. 16] allows to import a variety of 3D formats, including those of the main modeling software and some open formats such as Collada. For the export, the user can set the characteristics of the final product (lights, textures, views, etc.) and a customized page can be released [Fig. 17]. The 3D model displayed in the pdf viewer keeps segmentation – queryable with a layer menu, and the texture / color of the mesh. Operations like navigation, display types, measuring, etc. are all available.

Software: SimLab Composer pipeline:

- Import models from Blender through the format Collada (dae)
- assign links to the various parts of the model.
- export the 3D model by setting a template.

With respect to the open U3D2PDF solution, the final file size is quite large [Table 2].

Tab.2: Comparison of file sizes for the generated 3DPDF

PLY	Blender	3DPDF open source	3DPDF commercial
32552 KB	192313 KB	3985 KB	53813 KB

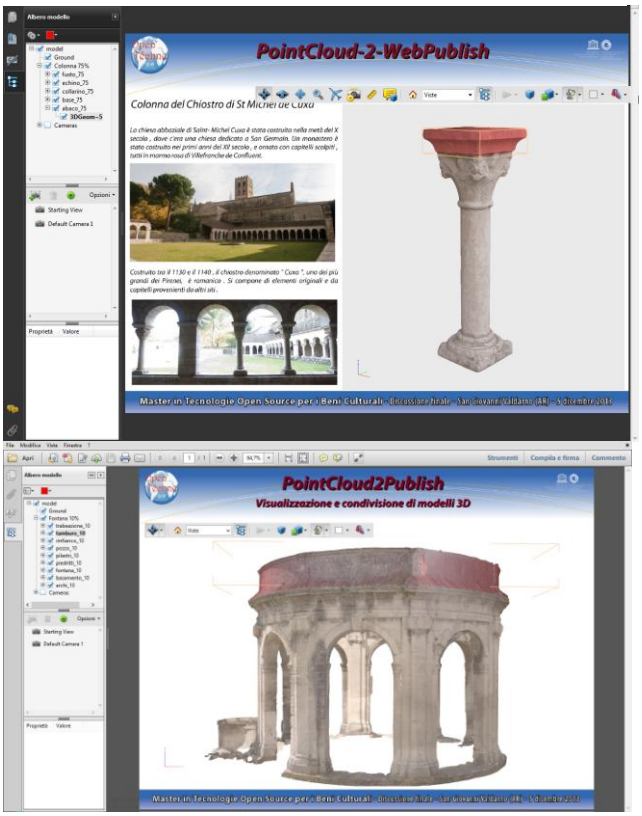


Fig. 17: Produced 3DPDF files by SimLab Composer.

3.5.2 WebGL

The visualization of 3D models within a web browser without installing any software or plug-in can be achieved exploiting WebGL (Web-based Graphics Library) in a context of HTML5. This solution has already been adopted widely by the 3D community [Manferdini and Remondino, 2011]. WebGL is currently regarded as part of the HTML5 standard, however, it is provided by the Khronos Group (a consortium of non-profit

sector) and not by the W3C 's organization that provides the standardization of HTML code. WebGL uses the hardware acceleration available in each computer directly within the browser. Most of actual web browsers support WebGL technology and using some ad-hoc code, web pages can be created to display 3D model online.

Among the available web-based solutions and repositories which allow to store and display 3D models, Sketchfab is probably the most promising [Fig. 18]. It's based on HTML 5 and WebGL and allows you to upload your 3D models on the web (downsampled to 50 MB if for free) and to share them privately or publicly. More than 20 3D file formats are supported. The visualization features are basically navigation, zooming, paning, rotation as well as measurements. The viewer keeps the texture, but not the eventual segmentation layers. 3D models cannot be download unless the owner allows it.

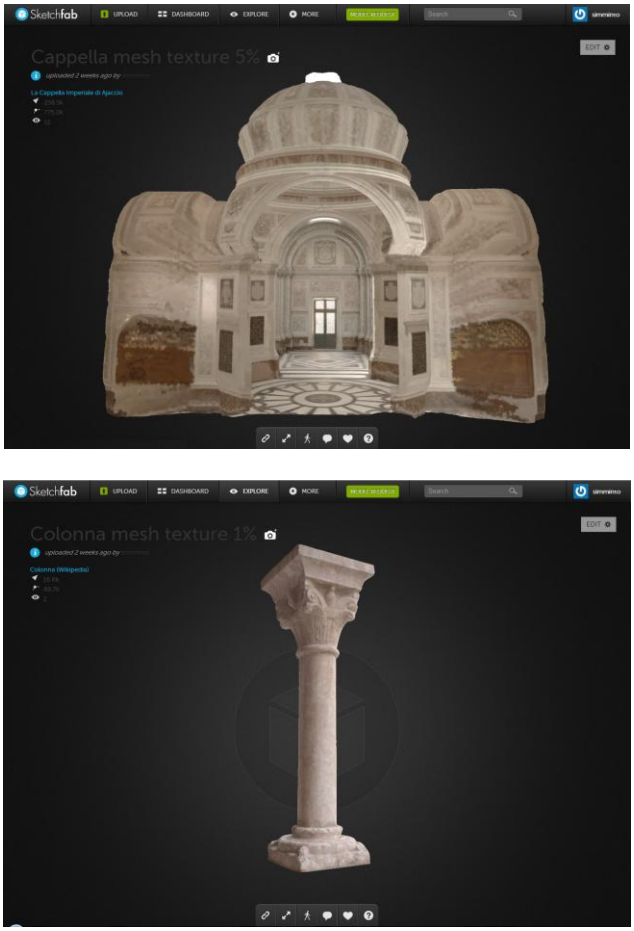


Fig. 18: Two examples of 3D model visualized in Sketchfab.

4. Conclusions

The paper reported the entire pipeline to publish and access a textured 3D model online, starting from an unorganized point cloud and producing a simplified mesh model which can be easily handle by non-experts on the web. Experiments and solutions, all perfomed with open source packages, were presented too.

Nowadays we must see the need to simplify and segment a 3D model to be able to publish it on the web as an opportunity to approach the study of the cultural property and to share our product with other researchers and users. Of course we need to coarefully consider IPR issues. The segmentation of a 3D model, following an architectural vocabulary, is an important approach that can be borrowed in other areas by adopting the vocabulary that best suits the application.

The web viewers are the last part of the chain but a very important feature as they must allow the user to easily interact with the produced 3D model, provide all the necessary details and information as well as preserve the possible segmention layers. The 3DPDF solution has limitations with large files, the geometry cannot be manipulated (IPR preservation), layers are well-kept and further textual information or links can be added. On the other hand, the WebGL solution needs some programming / coding expertise but some available experiences (e.g. http://www.3d.si.edu/browser) show the great potentialities of this approach.

For sure the generation of 3D models from real data (photogrammetry, 3D scanning, etc.) or CG methods will more and more flood our applications and 3D data will be ingresingly requested and accessed online. Solutions are already available but in the near future new developments will bring 3D models to everyone, everywhere, everytime.

REFERENCES

Di Benedetto M., Ponchio F., Malomo L., Callieri M., Dellepiane M., Cignoni P., & Scopigno R. (2014): Web and mobile visualization for cultural heritage. *3D Research Challenges in Cultural Heritage. A Roadmap in Digital Heritage Preservation*, LNCS Vol. 8355, (pp. 18-35), Springer-Verlag.

Berger M., Tagliasacchi A., Seversky L.M., Alliez P., Levine J.A., Sharf A., & Silva C. (2014). *State of the art in surface reconstruction from point clouds. Eurographics STAR*. In *Proceedings of EG'14*

Kazhdan M., Bolitho M., Hoppe H. (2006). Poisson Surface Reconstruction. In *Proceedings of 4th Eurographics symposium on Geometry processing*

Jiménez Fernández-Palacios B., Nex F., Rizzi A., Remondino F. (2014). ARCube - The Augmented Reality Cube for Archaeology. *Archaeometry*

Gallo G., Muzzupappa A., Bruno F. (2013). 3D reconstruction of small sized objects from a sequence of multi-focused images. J*ournal of Cultural Heritage*, 15(2), 173-182

Guidi G., Russo M., Angheleddu D. (2013). Digital reconstruction of an archaeological site based on the integration of 3D data and historical sources. *Int. Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 40(5/W1), 99-105

Landrieu J., Pere C., Rollier J., Catandet S., Schotte G. (2011). Digital rebirth of the greatest church of cluny maior ecclesia: from optronic surveys to real time use of the digital model. *Int. Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 38-5/W16, 31-37

Manferdini A.M., Remondino F., Baldissini S., Gaiani M., Benedetti B. (2008). 3D Modeling and semantic classification of archaeological finds for management and visualization in 3D archaeological databases. In *Proeedings of 14th Int. Conference on Virtual Systems and MultiMedia (VSMM)* (pp. 221-228).

Manferdini A.M., Remondino F. (2010). Reality-based 3D modeling, segmentation and web-based visualization. In *Proceedings of EUROMED2010 Conference,* LCNS Vol. 6436, (pp. 110–124), Springer-Verlag.

Menna F., Rizzi A., Nocerino E., Remondino F., Gruen A. (2012). High resolution 3D modeling of the Behaim globe. *Int. Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 39(5), 115-120.

Nony N., De Luca L., Godet A., Pierrot-Deseilligny M., Remondino F., van Dongen A., Vincitore M. (2012). Protocols and assisted tools for effective image-based modeling of architectural elements. In *Proceedings of EuroMed 2012 Conference*, LNCS Vol. 7616, (pp. 432-439) , Springer-Verlag.

Oehler B., Stueckler J., Welle J., Schulz D., Behnke S. (2011). Efficient Multi-Resolution Plane Segmentation of 3D Point Clouds. in *Proceedings of Intelligent Robotics and Applications*, LNCS vol. 7102, (pp. 145-156), Springer-Verlag.

Pierrot-Deseilligny M., De Luca L., Remondino F. (2011). Automated image-based procedures for accurate artifacts 3D modeling and orthoimage generation. *Geoinformatics FCE CTU Journal,* 6, 291-299

Pierrot-Deseilligny, M. (2014). MicMac, Apero, Pastis and Other Beverages in a Nutshell. Manual-Available at http://logiciels.ign.fr/?Telechargement,20 (Last access: December, 2014)

Potenziani M., Corsini M., Callieri M., Di Benedetto M., Ponchio F., Dellepiane M., Scopigno R. (2014). An advanced solution for publishing 3D content on the web. In *Proceedings of of Museums and the Web*

Remondino F., Campana S. (2014). 3D Recording and Modelling in Archaeology and Cultural Heritage - Theory and Best Practices. *Archaeopress BAR Publication Series,* 2598, 171 pp.

Remondino F., Menna F., Koutsoudis A., Chamzas C., El-Hakim S. (2013). Design and implement a reality-based 3D digitisation and modelling project. In *Proceedings of IEEE Conference "Digital Heritage 2013"*, Vol. 1, (pp. 137-144)

Remondino F. (2003). From point cloud to surface: the modeling and visualization problem. *Int. Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 34(5/W10)

Scopigno R., Cignoni P., Pietroni N., Callieri M., Dellepiane M. (2014). Digital Fabrication Technologies for Cultural Heritage. In *Proc. 12th Eurographics Workshops on Graphics and Cultural Heritage (EG GCH 2014)*, (pp. 75-85)

Stanco F., Battiato S., Gallo G. (2011). *Digital Imaging for Cultural Heritage Preservation - Analysis, Restoration and Reconstruction of Ancient Artworks*. CRC Press, 523 pp.