ACTA IMEKO
ISSN: 2221-870X
October 2018, Volume 7, Number 3, 64 - 72

A software tool for the semi-automatic segmentation of architectural 3D models with semantic annotation and Web fruition

Giorgio De Nunzio[1]

[1] *Dipartimento di Matematica e Fisica "Ennio De Giorgi", University of Salento, Lecce, Italy*

ABSTRACT
The thorough documentation of Cultural Heritage artifacts is a fundamental concern for management and preservation. In this context, the semantic segmentation and annotation of 3D models of historic buildings is an important modern topic. This work describes a software tool currently under development, for interactive and semi-automatic segmentation, characterization, and annotation of 3D models produced by photogrammetric surveys. The system includes some generic and well-known segmentation approaches, such as region growing and Locally Convex Connected Patches segmentation, but it also contains original code for specific semantic segmentation of parts of buildings, in particular straight stairs and circular-section columns. Furthermore, a method for automatic wall-surface characterization is devoted to rusticated-ashlar detection, in view of masonry-unit segmentation. The software is modular, so allowing easy expandability. It also has tools for data encoding into formats ready for model fruition by Web technologies. These results were partly obtained in collaboration with Corvallis SPA (Padua – Italy, http://www.corvallis.it).

Section: RESEARCH PAPER

Keywords: 3D models; semantic segmentation; photogrammetry; SfM; MVS; autocorrelation function

Citation: Giorgio De Nunzio, A software tool for the semi-automatic segmentation of architectural 3D models with semantic annotation and Web fruition, Acta IMEKO, vol. 7, no. 3, article 11, month year, identifier: IMEKO-ACTA-07 (2018)-03-11

Section Editor: Egidio De Benedetto, University of Salento, Italy

Received April 8, 2018; **In final form** August 23, 2018; **Published** October 2018

Funding: This work was partly supported by Corvallis SPA (Padua – Italy, http://www.corvallis.it)

Corresponding author: Giorgio De Nunzio, giorgio.denunzio@unisalento.it

1. INTRODUCTION

Nowadays, an important part of Architectural Heritage documentation consists in 3D models of buildings, obtained by photogrammetric techniques (i.e. from sequences of digital photos processed by Structure from Motion, SfM, and MultiView Stereo, MVS, procedures) or range data (by laser scanning). These models can be stored in databases to be retrieved via the World Wide Web. Professionals access this data for better Cultural Heritage (CH) comprehension, management and preservation, while tourists can take virtual tours before traveling, or use the models as an alternative to the usual souvenir photos. 3D digital representations can be semantically enriched by adding annotations (metadata), associated to the model as a whole or to its noteworthy elements. In order to partition a digital model into parts, a process known as segmentation is employed. When the parts are significant, i.e. they reflect a possible decomposition into standard architectural elements, and these elements are conveniently annotated, we obtain a semantic segmentation. As an example, an order [1] consists of three parts: the entablature, the columns, and the crepidoma or the pedestal; each component can be further subdivided, e.g. the entablature consists of the cornice, the frieze, and the architrave; the column is composed of the capital, the shaft, and may have a base, and so on. The partition process goes on until we obtain "atomic parts" which are the basic elements of classical architecture. Each of these parts has a name, it may have material information, a history, details on the conservation status, etc.

Fruition within the semantic Web requires semantics and annotations to be expressed in standard ways: metadata should be assigned by respecting field-specific ontologies. The standard formal ontology for CH is the CIDOC Conceptual Reference

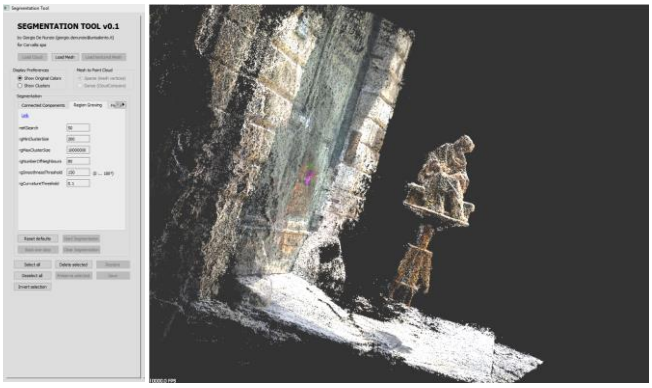

Figure 1. The software Graphical User Interface.

Model (CRM) (http://www.cidoc-crm.org/) by the International Council Of Museums (http://icom.museum), which provides definitions and a formal structure to describe the implicit and explicit concepts and relationships used in CH documentation. CIDOC-CRM extensions specific to Architecture exist, e.g. CRMba (http://www.cidoc-crm.org/crmba/home-7), created to support building documentation.

This article describes some results on the development of a software system for partial semiautomatic semantic segmentation, characterization, and annotation of building 3D models produced by photogrammetric surveys, and their fruition by Web technologies. The paper extends the work reported in [2].

After the Introduction, Section 2 reports some details on segmentation, while Section 3 gives a summary of annotation with Web fruition. The concluding section sums up the main results and states some perspectives.

2. SEGMENTATION

Many algorithms were integrated in the software to obtain the segmentation of buildings into their salient elements, such as stairs, columns (subdivided into capital, shaft and base), walls, doors, windows. The software is written in C++ with the Point Cloud Library, PCL, [3] and some Matlab parts for rapid prototyping and testing.

Several techniques and algorithms address the problem of point cloud or mesh segmentation, such as region growing, model fitting, and machine learning approaches [4]. It is well known that no single method is able to manage all the segmentation goals, so the software was designed with a modular approach, i.e. as a framework in which the implementation of new techniques might be easy. We used some ideas from the literature as well as our own algorithms, consistently integrating them in the system. Following this idea, a number of segmentation priorities were identified. Before all, some standard and generic segmentation approaches were implemented, to give the user a sort of "Swiss knife" to cope with different situations concerning object subdivision. Then, specific algorithms were inserted for particular tasks. Horizontal and vertical plane identification was achieved by RANSAC (RANdom SAmple Consensus) [5-8]: this step is preliminary to all the other segmentation tasks, because it allows the identification of walls and the floor, and the ceiling if present in the model. Wall identification is important in order to prepare for the recognition of openings (doors and windows) [9], to characterize wall-surfaces (e.g. distinguishing flat from rusticated walls), and to define the interior and the exterior of buildings. Stairs were identified by a generative approach, in which parametrically generated shapes are localized in the model by fitting procedures. Circular-section columns were recognized by ground histogram calculation. As a complementary result, walls were annotated as flat or rusticated by a method based on the autocorrelation function, and rusticated wall segmentation into the elementary wall element is planned in the near future.

The various segmentation processes are of course not completely automatic, because some parameters must be set in accordance with the particular model features and are not flawless.

In the following paragraphs, some more details and some results will be given concerning the overall software structure (Paragraph 2.1), the generic segmentation methods implemented (Paragraph 2.2), the detection and localization of straight stairs and round cross-section columns (paragraphs 2.3 and 2.4 respectively), the method for wall surface characterization (paragraph 2.5).

2.1. Overall software structure and user workflow

The software is structured in a modular way, providing some general-use segmentation methods, to be tested on typical architectural 3D models built out of a photogrammetric-reconstruction pipeline. The different methods will be definitely incorporated in the code based on the results gradually obtained during the tests. A picture of the software Graphical User Interface is shown in Figure 1.

Most existing segmentation algorithms work on point clouds. This software allows segmentation of both point clouds and (textured) meshes: in the latter case, the mesh is transformed into a "sparse" point cloud by "baking" the texture on the mesh vertices. It is planned to convert also the mesh to a "dense" cloud, by dense sampling. The segmentation algorithms are then applied to the derived cloud. After segmentation, if a particular cluster has been chosen as the desired result of the segmentation procedure, before saving to disk it is converted back to textured mesh.

The user workflow, to be adapted in various ways according to needs, is as follows:

a. load a 3D model/scene to be segmented (e.g. for annotation, or for model cleaning). The scene may be composed of different disconnected parts, and the various connected components may have a finer structure to be partitioned (e.g., a column could be segmented into a shaft, a base and a capital).
b. visually examine the 3D scene: thanks to the mouse-operated tools for space and object manipulation (panning, rotation, scaling, and viewpoint displacement), the scene is examined to identify the constituent parts of interest (and useless ones to be considered as noise, if present).
c. use the segmentation tools to cluster points and identify/characterize/label the connected components ("objects").
d. manipulate the objects of interest, with a variety of selection/deselection/deletion/saving tools (mouse click to explore and select, select all, deselect all, reverse selection, delete selected, delete unselected, restore deleted, save selected objects to disk)

e. start again with the segmentation procedures on one or more of the objects produced in the preceding steps.
f. if desired, add annotations to the objects.

One can imagine working on one object at a time, eliminating the others, segmenting them into parts, and saving the individual parts with their respective semantic labels: afterwards, it will be possible to reconstruct the whole scene by merging the labels and collecting the meshes together.

The procedure can be applied cyclically, selecting one or more clusters each time in order to interactively reach a useful and optimal segmentation.

An "undo" mechanism has been implemented to recover the previous state of clustering when the one obtained is not optimal: in this way, it is possible to try new algorithms or different values of the parameters

2.2. General-purpose point-cloud segmentation

At present, the following segmentation approaches, already available in PCL, have been tested and partially implemented in the software: Euclidean Cluster Extraction[1] (ECE), Conditional Euclidean Clustering[2] (CEC) Region Growing Segmentation[3] (RGS), Min-Cut Segmentation[4] (MCS), Difference of Normals Based Segmentation[5] (DoNS), Locally Convex Connected Patches[6] (LCCP). The ECE method implements a simple data clustering approach in a Euclidean sense by using a 3D grid subdivision of the space with fixed width boxes (octree). This data structure is very fast to build and is useful for situations where a volumetric representation of the point cloud is needed, or the data in each octree leaf should be approximated with a different structure. In ECE, the octree structure helps in the implementation of a flood-fill like clustering technique by efficiently determining each point nearest neighbours in a sphere with radius smaller that a given threshold. CEC is similar to ECE, with the advantage that the constraints for clustering are now customizable by the user. Some disadvantages include: no initial seeding system and no over- and under-segmentation control. RGS belongs to the same family; its purpose is to merge the points that are close enough in terms of a smoothness constraint computed by comparing the angles between point normals. The output is a set of clusters, where each cluster is a set of points that are considered a part of the same smooth surface. MCS is a well-known segmentation method [10] that creates a binary segmentation of the given input cloud. The algorithm divides the cloud into two sets: foreground and background points (points that are considered to belong to the object and those that are not). The algorithm is based on a graph representation of the data, in which the cloud points are the graph vertices, and edges connect the points to their nearest neighbours and to two special vertices called source and sink. Weights are assigned to the edges (smoothness cost) and foreground/background penalties are calculated. Finally, the search of the minimum cut is made and the cloud is divided into foreground and background. DoNS algorithm [11] performs a scale-based segmentation of the given input point cloud, finding points that belong to each cluster within the scale parameters given. It is a computationally efficient multi-scale approach, quite effective in the segmentation of scenes with a wide variation of scale. For each point in a point cloud, two unit point normals are estimated with different radii and the normalized (vector) difference of these point normals is calculated to define the DoN operator. The motivation behind DoN is the observation that surface normals estimated at any given radius reflect the underlying geometry of the surface at the scale of the support radius. Finally, LCCP [12] is an efficient unsupervised and model-less approach, which begins by breaking down the scene into a surface patch adjacency graph based on a voxel grid. The graph edges are then classified as convex or concave using a combination of simple criteria that work on the local patch geometry. In this way, the graph is divided into locally-connected convex subgraphs, which - with great accuracy - represent parts of the modelled object. In addition, a new depth-dependent voxel grid is used in order to address the decrease in cloud point density at large distances. This improves segmentation, allowing the use of fixed parameters for very different scenes. The algorithm is simple to implement and requires no training data, yet it produces results comparable to the state of the art of methods that incorporate high-level concepts such as classification, learning and model adaptation. The algorithm aims to separate parts of objects rather than selecting whole objects.

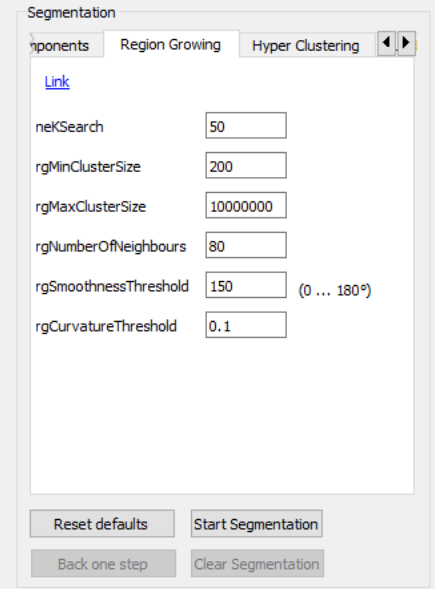

Figure 2. The control panel containing the tab widget that allows the choice of the segmentation algorithm and the assignment of parameter values.

In our software, a tab widget allows to choose between the various segmentation approaches, exposing the algorithm parameters to the user (see Figure 2). In Figure 3 some example of model segmentation are shown. Even if segmentation is applied and performed to the point cloud, the result is then projected onto the original textured mesh, so for example Figure 4 shows the statue separated from the support, saved as a textured mesh.

[1] http://www.pointclouds.org/documentation/tutorials/cluster_extraction.php

[2] http://pointclouds.org/documentation/tutorials/conditional_euclidean_clustering.php

[3] http://pointclouds.org/documentation/tutorials/region_growing_segmentation.php

[4] http://pointclouds.org/documentation/tutorials/min_cut_segmentation.php

[5] http://pointclouds.org/documentation/tutorials/don_segmentation.php

[6] http://docs.pointclouds.org/trunk/classpcl_1_1_l_c_c_p_segmentation.html

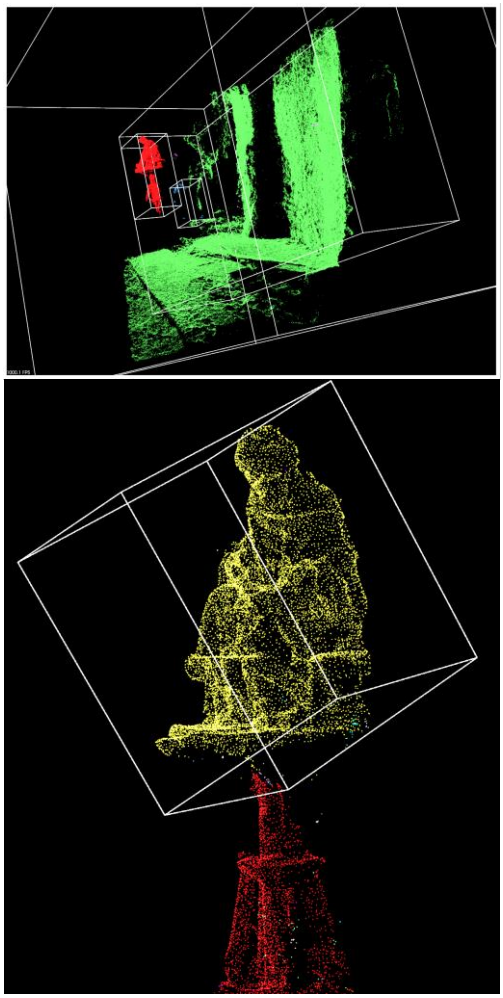

Figure 3. Examples of subsequent segmentation operations. Top: simple RGS was used to label the connected components with colour, thus separating the statue from the background. Bottom: LCCP allowed isolating the statue from its support. The white wireframe bounding boxes are used to select the various clusters prior to applying segmentation, preserving, deleting or saving commands.

2.3. Detection and segmentation of straight stairs

Stairs detection is important in several different fields, such as multi-storey path finding for explorer robots venturing into buildings [13–15], as an aid for the visually impaired [16–18], and last but not least in the semantic segmentation of 3D models of Architectural Heritage buildings [19, 20]. Algorithms developed for the first two cases are usually not directly applicable to semantic segmentation for CH, [15] because they generally work on organized point clouds derived from RGB-D data, while CH applications use unorganized clouds coming from photogrammetry or laser scanners. They may anyway suggest methods valid for unorganized point clouds too.

The literature proposes diverse techniques. Some start from edge point extraction. For example in [13] edge points are detected in RGB-D images by the Concave Hull algorithm available in the PCL framework, then classification relies on depth and geometric information. In paper [14], a staircase extraction algorithm based on super-voxels is proposed. In [15] the Authors develop a graph-based detection method for point clouds, which first segments planar regions and extracts the stair tread and riser segments. With these segments, a dynamic graph model is initialized that is used to detect stairs. See [15] also for a good review of some previous approaches useful for unorganized point clouds, and for a discussion on plane-based and edge-based methods. In [16] walls, doors, stairs, and a residual generic class of obstacles on the floor are detected in RGB-D data; stairs are found by searching for points on planes at increasing height from the ground, with a given step height and a tolerance. In [17] depth maps, calculated from RGB-D or stereo data, are used to feed a classifier. In [18] the Authors propose a staircase detection algorithm in RGB-D data, based on a support vector machine (SVM): the Hough transform is used to extract parallel lines in RGB frames so as to detect stairs candidates, and the depth frames are employed to classify the staircase candidates as upstairs, downstairs, and negatives (such as corridors).

Figure 4. The statue in Figure 3, after saving it as a fully textured mesh.

Paper [19] comes from the CH context and develops an application specific to unorganized point clouds, by a generative modeling approach: parameterized object models are built by GML, the Generative Modeling Language, and fitted to the point cloud (by subpart fitting without previous segmentation). In [20] an interactive framework to extract hi-level primitives (e.g. columns or staircases) from 3D models is presented, again based on a generative and fitting approach.

The particular approach described in this paper is inspired by [19]. It is generative and ideally automatic. The large computational effort, necessary for 3D subpart fitting is avoided by a pre-processing step which reduces the fitting problem to 2D. For this purpose, the horizontal straight edges outlining the steps are detected and their mean points are arranged into a 2D pattern used for fitting.

It is useful to introduce the method by first giving the appropriate terminology and by classifying the various types of stairs by geometrical considerations [21].

Stairs are made of series of steps (flights) with landings at appropriate intervals. Each step consists of a tread (the horizontal part, with its depth and width) and a riser (the vertical part between treads, with its height). We can distinguish: straight stairs (consisting of either one single flight or more than one flight with landings, with no change in direction; if there is a change in direction, we have parallel or angle stairs, such as quarter-turn stairs, half-turn or dogged-legged stairs, etc); circular stairs; spiral stairs, and others. Sometimes only stairs without direction change are considered as true straight stairs.

Our algorithm only addresses straight stairs, with or without direction change, so that we can assume parallel stairs edges.

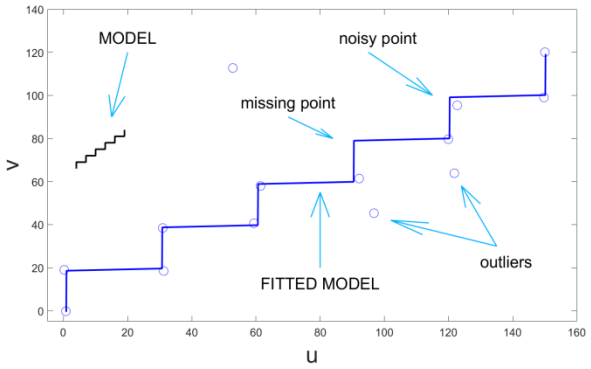

Figure 5. Point-pattern fitting. The small circles are the real-world points obtained by projecting the edges centres (in an edge cluster) onto a plane perpendicular to the edges (step 4; *u* and *v* are the coordinates in the projection plane). There are some outliers (corresponding to erroneous edges), points are noisy, and some may be missing. The model is on the left. The fitted model perfectly reconstructs the stairs silhouette.

As mentioned, a parametric generative model was chosen for its versatility. The problem complexity was reduced by projection to a 2D space before model fitting. As illustrated in detail afterwards, the algorithm first detects the most significant edge points in the cloud (Step 1) and finds the straight edges by RANSAC (Step 2); then the horizontal edges are preserved and clustered into groups of parallel segments (Step 3); the clusters are projected to a plane perpendicular to the segments, so getting a 2D pattern of points representing the cluster, and modelling the stairs (Step 4); finally a parameterized generative model of stairs is fitted to the point pattern (Step 5), giving an accuracy measure and the model parameters; if the fit accuracy is satisfying, the position of the stairs in the original 3D world is found, and a bounding box is visualized in the cloud. In more detail:

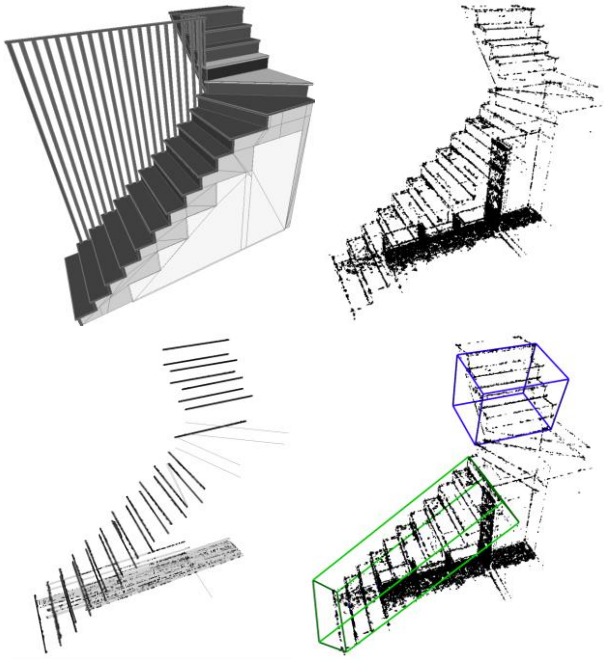

Figure 6. A case study. The stairs model (top-left picture) was downloaded from https://archive3d.net (model N290716) and converted to a point cloud by the CloudCompare software (http://www.cloudcompare.org/), the edge points were detected (top-right image), then the straight edges were located and the horizontal ones were clustered (bottom-left picture; light-grey edges are those rejected as outliers). The straight stairs were then detected as explained in the text, and automatically highlighted with bounding boxes (bottom-right picture).

Step 1: edge points are detected by the fast and precise method described in [22], where sharp edge features are discovered by analysing the eigenvalues of the covariance matrix that are defined by each point's *k*-nearest neighbours. The C++/PCL code for edge detection is kindly shared by the Authors.[7]

Step 2: straight lines in the edge point cloud are detected by the RANSAC method.

Step 3: the edges found at Step 2 are clustered by DBSCAN[8] (Density-Based Spatial Clustering of Applications with Noise) [23], a density-based clustering algorithm which can potentially identify clusters of any shape in a data set containing noise and outliers. DBSCAN usually clustered the stairs edges very well, collecting in a same group almost all and only the detected edges pertaining to a single flight of stairs. Other clustering algorithms were tested with far inferior results. The features used for clustering were the stair direction (one of the two non-zero direction cosines of the edges) and the three coordinates of the edge center. A larger weight was given to the direction feature because only strict parallel edges could belong to the same stairs, having decided to limit to straight stairs.

Step 4: for each cluster, a plane is now defined, normal to the cluster edges, and the edge centers are projected onto it, so giving a planar pattern of points representing the possible stairs. The coordinates of these points on the plane are defined by introducing a pair of basis vectors ($\boldsymbol{u}$, $\boldsymbol{v}$) (orthogonal and unit length). The definition of $\boldsymbol{u}$ and $\boldsymbol{v}$ is not unique, but a convenient way is the following. Let $\boldsymbol{n} = (a, b, c)$ be the plane normal unit vector: in our case, a is the average x direction cosine of the edges in the cluster, $b \sim 0$ (y being the vertical axis, and the edges being horizontal), and $c = \mathrm{sqrt}(1 - a^2)$. We can set:

$$\begin{aligned} \boldsymbol{u} &= (c, 0, -a) \\ \boldsymbol{v} &= \boldsymbol{n} \times \boldsymbol{u} \end{aligned} \tag{1}$$

where the × symbol represents cross product.

We now project each edge center (x_c, y_c, z_c) onto the $(\boldsymbol{u}, \boldsymbol{v})$ plane and calculate its (u_c, v_c) coordinates by dot product:

$$\begin{aligned} u_c &= \boldsymbol{u} \cdot (x_c, y_c, z_c) \\ v_c &= \boldsymbol{v} \cdot (x_c, y_c, z_c) \end{aligned} \tag{2}$$

where the origin of the (u, v) coordinates is the world origin (0, 0, 0). Each point ideally comes from an edge, so marking the transition from riser to tread, or from tread to riser. This pattern can contain noise (points due to incorrect edges) and may miss some points related to edges that were not found in Step 2.

Step 5: a point pattern is generated as a stairs model (each point being the projection of a stairs edge), depending on the following stairs parameters: number of steps, riser height, tread depth, 3D stair position (i.e. the position of the centre of first edge of the lowest stairs step). The model is fitted to the patterns obtained in Step 4, and the optimized parameters, as well as the value of the objective function as an accuracy measurement, are returned. These values are then automatically

[7] https://github.com/denabazazian/Edge_Extraction

[8] http://yaikhom.com/2015/09/04/implementing-the-dbscan-clustering-algorithm.html

evaluated in order to accept or reject the cluster as representing stairs. The point fitting algorithm, which gave the best results in terms of robustness and accuracy, was Coherent Point Drift (CPD) [24]. Figure 5 shows an example of model matching.

After stairs detection, the model consists of the following parameters: lower-step center coordinates, stairs direction parameter in the horizontal x-χ plane, the number of steps, tread depth, riser height, and step width.

The procedure was tested on synthetic and real-world models. Step 5, in particular, was tested against synthetic point patterns, and the fitting procedure was stressed by adding noise to the ideal point position, by inserting points not corresponding to stairs edges, and by removing significant points.

Figure 6 shows the application of the algorithm to a synthetic case of study.

2.4. Detection and segmentation of round cross-section columns

The literature is not rich in papers concerning column detection in 3D scenes. For example, ef. [18] applies generative models also to columns and arcades, while in [25] round and rectangular cross-section columns are detected by ground histogram calculation. For the sake of performance, we chose the latter approach, which decreases the problem dimensionality from 3D to 2D, and decided to initially limit the scope to circular columns.

The model point cloud was projected to the ground plane, and planar point density was evaluated, so obtaining ground histograms. The vertical structures left easily recognizable signatures in the point-density map, e.g. circular columns produced circular shapes that could be located by the Circular Hough transform (CHT), [26] with subsequent column detection and segmentation. The columns were then partitioned into their constituent parts. The details follow.

Step 1: after down-sampling, the point cloud is projected to the ground plane.

Step 2: with the purpose of calculating point density in the plane, and finding the signature of vertical structures, an octree is built from the projected cloud, and box-searching is performed in each cell so as to measure each cell occupancy; an image is then built, containing one pixel per octree cell, with grey value proportional to the point density in that cell.

Step 3: CHT is applied to the image created in Step 2, to find the signatures of columns (circles) if present.

Step 4: the centers and radii of the circles give hints on the horizontal column location, but nothing is known about vertical position and size: therefore, a region, centered on each column center and slightly larger than the column diameter, is searched for a cylinder by RANSAC; looking for cylinders only in the regions located by the CHT reduces false positives and speeds thing up.

Step 5: the columns are finally segmented into their constituent parts (capital, shaft and base) by convex decomposition [27]; this algorithm computes a hierarchical segmentation of the mesh triangles by applying a set of decimation operations to the mesh dual graph, guided by a cost function related to concavity: the generated segmentation is then exploited to construct an approximation of the original mesh by a set of convex surfaces; the generated convex hulls can be used as space filters to segment the original points of the point cloud. Segmentation is not always accurate enough, causing small segments under the column base or in the architrave – beside the three main column parts – to be returned. To overcome this problem, the longest segment is identified and assigned to the shaft, and the two segments connected to it from above and from below are selected as the capital and the base respectively, disregarding the other parts if present. Of course, segmentation quality is affected by the model quality.

Figure 7. Top (black background): the Apollo theatre, in Lecce (Italy). The model is followed by the ground point density map (obtained by projecting the model to the ground plane), where CHT (Circular Hough Transform) was applied, which allows column detection: the column signatures are shown in red). A picture of the isolated columns follows. Bottom (white background): the Gerrard Hall (North Carolina) model, after column detection and segmentation into capital, shaft, and base (as convex component parts). The model is followed by the ground density map with CHT applied.

Figure 7 shows the application to a model of the Apollo Theatre (Lecce, Italy) and to the Gerrard Hall (North Carolina) both reconstructed by photogrammetry from sets of about 100 pictures each (Gerrard Hall photos are available at https://colmap.github.io/datasets.html). See the figure caption for details.

2.5. Wall-surface characterization: flat vs rusticated walls

This module is not a segmentation tool in itself. Rather, it acts as an automatic classification/annotation instrument and as a pre-processing step for rusticated-ashlar masonry-unit segmentation. For this reason, it is associated with the wall segmentation module.

Two basic patterns used in stone masonry walls are rubble and ashlar masonry [28-31]. The former is made from stones with irregular sides (not at right angles to each other), while in the latter, the stone sides are dressed square (at right angles to each other). The front and back faces of the stone may be dressed or undressed. Both rubble and ashlar masonry may be random or coursed, according to mortar joint regularity. Another attribute of masonry is rustication, in which the visible faces of the dressed stones are raised with respect to the horizontal and usually the vertical joints, which may be rabbeted, chamfered, or bevelled. The joints are therefore emphasized. The stones may be left rough or worked in such a way as to give a strong textural effect. In particular, rusticated ashlar is a coursed and dressed stonework with edges and joints emphasized by rustication. Rustication has been used since ancient times and was first described by Serlio in his famous treatise [32]. One of the most known rusticated surfaces belongs to the Palazzo dei Diamanti (Diamond Palace) in Ferrara, with its rusticated façade of diamond-shaped marble blocks.

The purpose of this software module is to distinguish various types of stonewalls, in particular discriminating rusticated ashlar from "flat" masonry.

From a mathematical point of view, a rusticated-ashlar masonry cloud point is a surface presenting some sort of periodicity. Let the x and the y axis be respectively horizontal and vertical, then the joints parallel to the x axis are always rearward with respect to the stone face, and many times this is true also for the joints parallel to the y axis. As a consequence, periodicity appears in the y axis or in both.

Periodicity can be studied and characterized by a large number of tools, such as Fourier analysis, the autocorrelation function, wavelets, and others. In this work, the Fourier transform and the autocorrelation function were tried, and the latter proved simpler, effective, and more resistant to noise. This approach will be briefly described here, and some results will be given.

The autocorrelation function (ACF) for a real, continuous, univariate function $f(x)$ is defined as:

$$a(t) = \int_{-\infty}^{+\infty} f(\tau)\, f(t+\tau)\, d\tau \tag{3}$$

This definition, in the case of real, discrete, bivariate functions on a finite support such as 2D digital images I, becomes:

$$a(x,y) = \sum_{m=1}^{M} \sum_{n=1}^{N} I(m,n)\, I(x+m, y+n) \tag{4}$$

The ACF can be used as the basis of a texture measure in 2D images, because peaks in the autocorrelation function of a regular-texture image characterize the texture periodicity. Compared with the peaks in the Fourier spectrum, the peaks in the ACF are usually more prominent and thus easier to find. Moreover, the ACF is also less disturbed by noise. The use of the ACF for 2D-image texture characterization is well-established [33].

In the scope of this paper, the interest is not in 2D images, but in 3D point clouds of building walls. It is nonetheless very easy to convert our 3D problem to a 2D one, considering that a rusticated surface can usually be expressed as a (non-uniformly sampled) 2D function $h = f(x, y)$ where h is the stone point elevation from a reference plane xy parallel to the wall, and the (x,y) coordinates locate the point projection to the plane. Assigning a grey level to h, and applying (e.g. bilinear) 2D interpolation, the wall can be converted to an image and all the tools available for grey-level 2D images can be employed. This is totally different from taking a single photograph of the surface, which records colour and shadows but not directly 3D structure.

So, in this work, the wall surface was converted to an image, the ACF was calculated, and the horizontal and vertical profiles (1D sections) passing through the ACF center were derived. These profiles respectively contain the information on x and y periodicity, if present, as coded in a certain number of peaks. The absence of periodicity gives only a central peak, while periodic signals give secondary peaks prominent from the background. These peaks are smaller than the central one but evident.

In order to get rid of the slowly varying ACF background, a 1D top-hat morphological operation is applied to the profiles, and the central peaks are measured (for intensity reference) and removed. Finally, the height of the remaining peaks is measured. The ratio between the mean height of the latter peaks, and the central-peak height, is used as a feature to assess horizontal or vertical periodicities, which correspond to the presence of rusticated ashlar. If a profile only contains the central peak, 0 is assigned to the corresponding periodicity feature. Large values (> 0.1) are a sign of periodicity. The result of periodicity assessment is finally associated as an annotation to the wall under exam.

In Figure 8 two cases are shown, a rusticated façade and a flat wall.

In perspective, a similar approach can be used to estimate a representative tile of the rusticated ashlar masonry, or to study stone damage. A specific paper concerning method sensitivity, robustness to noise, and accuracy as a function of feature threshold, is in preparation.

3. SEMANTIC ANNOTATION AND WEB FRUITION

The system allows to insert annotations into the segmented model according to standard CIDOC-CRM entity E22 (Man-Made Object) and property P46 (is composed of), for future compatibility with the semantic Web. The models are then converted to the Nexus file format [34, 35]. Nexus, being a progressive-mesh format, allows smooth LOD (Level of Detail) choice depending on the current view (i.e., model distance, framing, device power). With this approach, it is possible to initially display a model with the lowest LOD and then let it gradually show more and more details, with effective real time visualization. As to Web fruition, our HTTP pages are dynamically built based on 3DHOP [36] for real-time model drawing, and some JavaScript code for semantic annotation visualization by an information tree linked to the model: a click on the model hotspots, defined by the previously segmented architectural elements, allows selecting the corresponding tree node, and vice versa (see Figure 9).

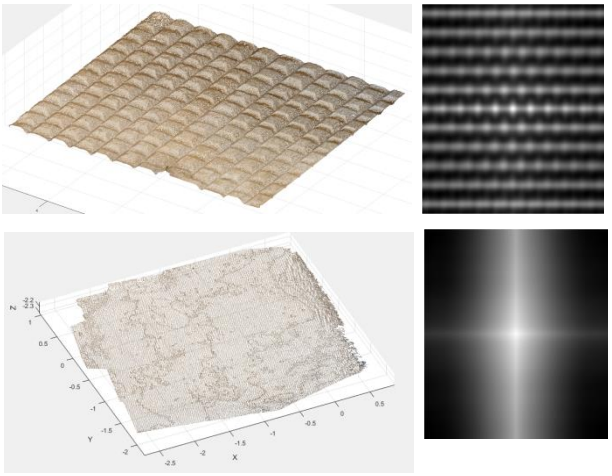

Figure 8. Two cases of wall periodicity measurement by the autocorrelation function, for rusticated ashlar masonry detection. Top: a rusticated wall (interior of Palazzo Adorno – Lecce, Italy); bottom, a flat wall with noise (slight stone degradation). Both cases show the point cloud (left) and the ACF of the corresponding 2D grey-level image obtained from the point cloud (right). The rusticated wall gives well-defined peaks in the ACF, related to periodicity, while the flat façade has none but the central one. In the former case, the value of the periodicity features calculated as described in the text is around 0.4, while in the latter case they are lower than 10^{-3}.

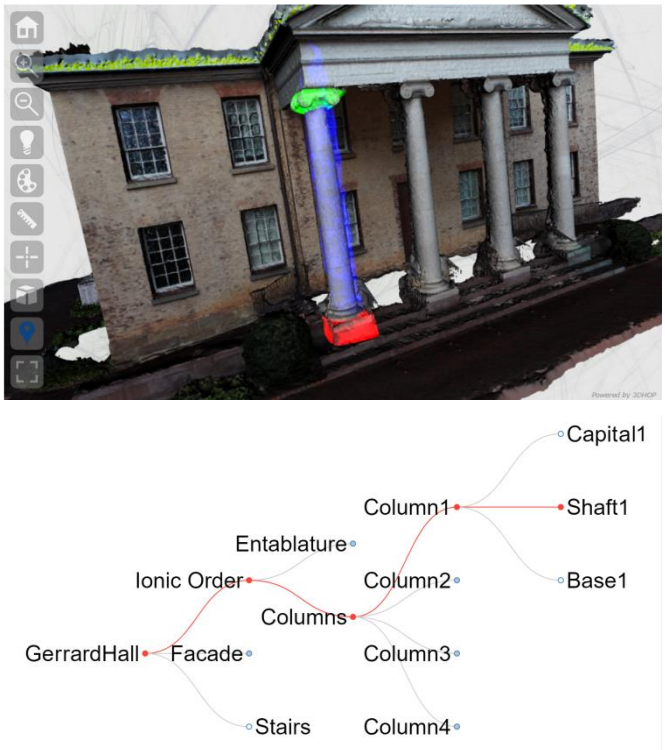

Figure 9. The Gerrard Hall model inserted in the Web system prototype, using 3DHOP and some JavaScript code for tree management. Hotspots were inserted at the column parts (top) and used as links to the semantic-structure tree (bottom). Clicks on the model allow navigating the tree.

CONCLUSIONS

The paper presents a software framework for semantic segmentation and Web fruition of Cultural Heritage 3D models. The software includes some standard general-purpose segmentation approaches available in PCL, and some specific semantic segmentation modules (in particular, straight stairs and circular-section columns). It also contains modules devoted to object characterization, such as the one currently under test, for the detection of rusticated ashlar masonry. In addition, the paper gives some details on the Web system prototype. The software proved successful in a number of monument models and synthetic point clouds. Nonetheless, the framework is still under development, with many segmentation modules relying on parameter tuning. The development will continue, with new modules added to the framework, and limitations hopefully relaxed.

ACKNOWLEDGEMENT

The results reported in this work were partly obtained in collaboration with Corvallis SPA (Padua – Italy, http://www.corvallis.it), who also financially supported the research.

REFERENCES

[1] A. Tzonis, L. Lefaivre, Classical Architecture: The Poetics of Orders, 1987, The MIT Press.

[2] G. De Nunzio and M. Donativi, Semi-automatic segmentation of architectural 3D models with semantic annotation and Web fruition, Proc. IMEKO International Conference on Metrology for Archaeology and Cultural Heritage, Lecce, Italy, October 23-25, 2017.

[3] R. B. Rusu and S. Cousins, 3D is here: Point Cloud Library (PCL), IEEE Int. Conf. on Robotics and Automation (ICRA), May 9-13, 2011, Shanghai, China.

[4] E. Grilli, F. Menna, F. Remondino, A review of point cloud segmentation and classification algorithms, The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. XLII-2/W3, 2017, 3D Virtual Reconstruction and Visualization of Complex Architectures, 1–3 March 2017, Nafplio, Greece.

[5] M. A. Fischler and R. C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Communications of the ACM, vol. 24, issue 6, June 1981, pp 381-395, doi:10.1145/358669.358692.

[6] P. Tang, D. Huber, B. Akinci, R. Lipman, A. Lytle, Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. Automation in Construction Vol 19, 2010, pp 829–843.

[7] K. Chen, Y.-K. Lai, S.-M. Hu, 3D indoor scene modeling from RGB-D data: a survey, Computational Visual Media Vol. 1, No. 4, December 2015, 267–278.

[8] V. Pătrăucean, I. Armeni, M. Nahangi, J. Yeung, I. Brilakis, C. Haas, State of research in automatic as-built modelling, Advanced Engineering Informatics, 29 (2), 2015, pp. 162-171, ISSN 1474-0346, http://dx.doi.org/10.1016/j.aei.2015.01.001.

[9] A. Martinović, J. Knopp, H. Riemenschneider, L. Van Gool, 3D All The Way: Semantic Segmentation of Urban Scenes From Start to End in 3D, 2015 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).

[10] A. Golovinskiy, T. Funkhouser, Min-Cut Based Segmentation of Point Clouds, Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on, 27 Sept.-4 Oct. 2009, Kyoto, Japan, DOI: 10.1109/ICCVW.2009.5457721.

[11] Y. Ioannou, B. Taati, R. Harrap, M. Greenspan, Difference of Normals as a Multi-Scale Operator in Unorganized Point Clouds, 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on, 13-15 Oct. 2012, Zurich, Switzerland, DOI: 10.1109/3DIMPVT.2012.12.

[12] S. C. Stein, M. Schoeler, J. Papon and F. Wörgötter, Object Partitioning using Local Convexity, Proc. of 2014 IEEE

Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 304-311.
[13] L. A. V. Souto, A. Castro, L. M. G. Gonçalves, T. P. Nascimento, Stairs and Doors Recognition as Natural Landmarks Based on Clouds of 3D Edge-Points from RGB-D Sensors for Mobile Robot Localization, Sensors 17, 1824; 2017.
[14] K.-W. Oh, K.-S. Choi, Supervoxel-based Staircase Detection from Range Data, IE Transactions on Smart Processing and Computing, vol. 4, no. 6, 2015.
[15] T. Westfechtel, K. Ohno, B. Mertsching, D. Nickchen, S. Kojima, S. Tadokoro, 3D Graph Based Stairway Detection and Localization for Mobile Robots, 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon Convention Center, October 9-14, 2016, Daejeon, Korea.
[16] H.-H. Pham, T.-L. Le, N. Vuillerme. Real-Time Obstacle Detection System in Indoor Environment for the Visually Impaired Using Microsoft Kinect Sensor, J Sensors, Vol. 2016, Art. id 3754918.
[17] S. Cloix, G. Bologna, V. Weiss, T. Pun, D. Hasler, Low-power depth-based descending stair detection for smart assistive devices, EURASIP Journal on Image and Video Processing, 2016, 2016:33.
[18] R. Munoz, X. Rong, Y. Tian, Depth-aware indoor staircase detection and recognition for the visually impaired, 2016 IEEE Int. Conf. on Multimedia & Expo Workshops (ICMEW), 11-15 July 2016.
[19] T. Ullrich, V. Settgast, D.W. Fellner, Semantic Fitting & Reconstruction, ACM Journal on Computing and Cultural Heritage 1 (2), 2008.
[20] D. Portelli, F. Ganovelli, M. Tarini, P. Cignoni, M. Dellepiane, R. Scopigno, A framework for User-Assisted Sketch-Based Fitting of Geometric Primitives, Proc. of WSCG, 18th Int. Conf. on Computer Graphics, Visualization and Computer Vision, 2009.
[21] S.D. Edgett and A.M. Williams, Vertical Circulation, in: Merritt FS and Ricketts JT (editors), Building design and construction handbook, 6th edition, McGraw-Hill, ISBN 0-07-041999-X (Section 16), 2001.
[22] D. Bazazian, J. R. Casas, J. Ruiz-Hidalgo, Fast and Robust Edge Extraction in Unorganized Point Clouds, Proc. of International Conference on Digital Image Computing: Techniques and Applications (DICTA), 2015.
[23] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise. Proc. of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96). AAAI Press, 1996, pp. 226–231. ISBN 1-57735-004-9.
[24] A. Myronenko, X. Song, Point-Set Registration: Coherent Point Drift, IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 32, issue 12, 2010, pp. 2262-2275.
[25] L. Díaz-Vilariño, B. Conde, S. Lagüela, H. Lorenzo, Automatic Detection and Segmentation of Columns in As-Built Buildings from Point Clouds, Remote Sens. 2015, 7, 15651-15667.
[26] R.O. Duda, P.E. Hart, Use of the Hough Transform to Detect Lines and Curves in Pictures, Communications of the ACM 15, 1972, pp. 11– 15.
[27] K. Mamou, and F. Ghorbel, A simple and efficient approach for 3D mesh approximate convex decomposition, Proc. of 16th IEEE International Conference on Image Processing (ICIP), 2009.
[28] N. Davies, E. Jokiniemi, Dictionary of Architecture and Building Construction, Elsevier, 2008, ISBN: 978-0-7506-8502-3.
[29] F. D. K. Ching, Building Constructions Illustrated 5th ed., Wiley, 2014 , ISBN 978-1-118-45834-1, p. 5.33.
[30] J. Summerson, The Classical Language of Architecture, revised and enlarged Ed., 1980, Thames and Hudson, ISBN 0-500-20177-3, p.45–47, p. 58–59.
[31] R. Chitham, The Classical Orders of Architecture, 2nd ed, Elsevier, 2005, ISBN 0 7506 61240, p. 190-192.
[32] S. Serlio, "Libro Quarto – Regole Generali di Architettura di Sebastiano Serlio Bolognese sopra le cinque regole degli edifici" XVI (1537).
[33] H.-C. Lin, L.-L. Wang, S.-N. Yang, Extracting periodicity of a regular texture based on autocorrelation functions, Pattern Recognition Letters 18 (1997) 433-443.
[34] P. Cignoni, F. Ganovelli, E. Gobbetti, F. Marton, F. Ponchio, R. Scopigno, Batched Multi Triangulation, Proc. IEEE Visualization, pp 207-214, October 2005.
[35] F. Ponchio, M. Dellepiane, Fast decompression for web-based view-dependent 3D rendering, Web3D 2015. Proc. of the 20th International Conference on 3D Web Technology, pp 199-207
[36] M. Potenziani, M. Callieri, M. Dellepiane, M. Corsini, F. Ponchio, R. Scopigno, 3DHOP: 3D Heritage Online Presenter, Computers & Graphics Volume 52, November 2015, Pages 129-141.