Volume xx (200y), Number z, pp. 1–6

Recovering 3D architectural information from dense digital models of buildings

Andrea Spinelli, Fabio Ganovelli, Claudio Montani, Roberto Scopigno [†]

Abstract

In recent years the progress of 3D scanning technologies and the consequent growing commercialization of scanners opened a large spectrum of opportunities for many professionals. In particular, architects and engineers may access to a digital model of a building without having to model it using a CAD software. On the other hand, there are two important differences between the digitized and the handcrafted model. The first is the absence of interpretation. The digitized model is only a set of polygons that describe, possibly in a very accurate manner, the scanned object. It does not provide the user with any other information about what a surface is (a wall, a window, an arch etc.) that, conversely, can be incorporated during the editing in a CAD session. The second difference is excess of realism. In the digitized models the are no planar walls, no right angles, no straight edges, simply because they are not, at the millimetric scale. Unfortunately, if a model must be used in a FEM simulation, for example, a CAD like model would be required. This paper describes an application framework and some techniques that have been implemented to help a non computer-graphics user in handling digital models of buildings acquired using 3D scanning. The techniques permit to visualize efficiently the models independently from their size, recover 3D information (measurements, sections, ...), extract geometric features and fit high level geometric primitives.

Categories and Subject Descriptors (according to ACM CCS): I.4.8 [Scene Analysis]: Range data
Keywords: 3D scanning, feature extraction, data fitting.

---

1. Introduction and previous work

The increasing availability of 3D range scanning devices, the development of software more and more efficient and user-friendly for the creation and manipulation of complex 3D digital models and, last but not least, the drop of the scanning technology costs are the main reasons of the recent fast proliferation of scanning campaigns for the acquisition of the shape of real objects.

The application fields taking advantage of 3D scanned data range from the reverse engineering, for the reconstruction of CAD digital models from physical objects, to the cultural heritage, for the documentation and fruition of 3D works of art, from the entertainment industry, to speed up the creation of cartoon films, to the medicine, with particular uses in dermatology and dentistry.

3D range scanning is more and more used in architecture as well; as a matter of principle the 3D cloud of points (or the corresponding 3D triangle mesh) representing the surface of a building contains all the information an architect needs in order to design hers/his operation. Moreover, the way to obtain such information is easier and faster than the classic methods based on theodolite or total stations.

Actually, the situation is not so bright: professionals turn out to have a lot of difficulties in the efficient usage of 3D scanning data for, at least, three main reasons:

- the size of the 3D dataset is often very large and this limits the importing of the data into the more used and known CAD systems;
- the 3D range scanning data, organized in clouds of points or triangle meshes, describe the surface of the investigated object but nothing of the available information id directly useable by a professional. The data do not present edges, nor project lines, nor geometric features, nor high level geometric primitives;
- the architectural design is based on the use of exact and measurable geometric primitives. It's somewhat paradoxical to observe that the *excess of realism* of the 3D range

---

[†] Visual Computing Laboratory, ISTI-CNR, Pisa (I) email: name.surname@isti.cnr.it

data represents a limitation for efficient measurements: no planar walls in the digital model, no right angles, no straight edges, and so on.

For these reasons, the application software we are developing at the Visual Computing Lab and which is summarily described in this paper tries to give a complete answer to these open problems and to fill the gap between the architectural 3D range scanning and an efficient use of the data by the professionals.

Our solution permits the user:

- to visualize and interact with a 3D digital model in the form of a triangle mesh. The size of the mesh is not a priori limited;
- to perform linear and surface measurements, to recover sections by intersecting the model with a single plane or a stack of parallel planes;
- to draw 3D geometric primitives directly on the model. The drawing of a segment onto the screen causes the creation of the corresponding 3D open polygonal line which leans against the model and represents the projection of the 2D segment back in the 3D space;
- to automatically recover lines and edges from the model and to fit high level known geometric primitives in an assisted way. Typical examples of high level primitives are arches and windows but the system can easily incorporate new user defined primitives.

Clearly the program enables to export the recovered information in different output formats and to generate high resolution snapshots of the scene.

The tackled problems are not new and there exist many different solutions in the scientific literature. However, the existing solutions do not face the problem in an integrated manner but they try to solve one or more aspects of the problem itself.

Let us examine some of the main aspects.

The 3D range scanning of buildings, archeological sites, or large industrial implants is generally carried out by means of time-of-flight laser scanning devices. The time-of-flight technology [CM02] permits to acquire in reasonable times (about 1000 samples per second) and good accuracy (6-8 millimeters with 8-10 millimeters of spatial resolution) the description of the surface of buildings. The typical working distance (device-target) is between 20 and 100 meters.

The 3D range scanning is not the only available technology for the recovering of 3D information of buildings [BSZF99,SB03]. The classic photogrammetric approach can be extended by means of model-based reconstruction methods in order to obtain the detailed reconstruction from close-range images. The 3D points obtained through image matching are generally segmented into a coarse polyhedral model with robust regression algorithm, then the geometry of the model is refined with predefined shape templates in order to automatically recover a CAD-like model of the building surface. Reprojection of the 3D shape templates is used to optimally fit their parameters to the image information.

These methods do not require large financial investments but they are unsatisfactory in terms of usability and amount of the recovered information. In our work, we use 3D range scanning data acquired by means of a time-of-flight laser device. Starting from a dense and huge cloud of points, we use the Visual Computing Lab's software tools to generate a 3D digital model [CCG*03]. Ad hoc data structures [CGG*04] permit to visualize and interact with the 3D meshes of the model without caring about the size (in terms of vertices and faces) of the meshes themselves.

In order to make the use of the 3D model more efficient and to reduce the amount of information to be handled, heavy-duty geometric simplification techniques can be applied. The existing solutions are numerous and the results are really good [CMS97, Lue01, LT99].

Unfortunately, the geometric simplification of the 3D model of our building is not adequate for the recovering of information *useful* for the professional in a CAD environment. In fact, the geometric simplification algorithms aim at removing redundant information and replacing sets of (quasi) planar adjacent triangles with larger ones. This ensures the lightening of the model but not the characterization of the interesting geometric features.

A radical improvement towards an efficient use of 3D architectural range data can be only obtained by means of feature extraction techniques.

Feature extraction from 3D data is unlike feature extraction in 2D images. Extraction in images relies on colour or intensity information that gives important clues about features (e.g. lines). In contrast, a pure 3D point cloud or triangle mesh contains no colour information, but three dimensional structural features (e.g. 3D lines) are inherent in its data. However, feature extraction within an irregular 3D point cloud or triangle mesh from 3D scanning is difficult, mainly because it is not obvious if we are in front of a change of the curvature corresponding to the separation lines between two adjacent features rather than local noise which is a common drawback in the time-of-flight laser range scanning.

Features extraction techniques can be broadly classified into two large groups: (a) the automatic features extraction from the 3D triangle mesh or cloud of points (in practice the automatic location of the lines separating homogeneous regions of the surface) and (b) the fitting to the 3D data of high level geometric primitives. In this case, the user chooses the primitive to be fitted (arches, cylinders, boxes, etc.) while the system computes the corresponding geometric parameters.

As previously mentioned, the techniques of the first group suffer from the noise of the data in the case of 3D range scanning. Many authors tried to surpass this serious drawback

[OBS04, WB01] but the best results are probably obtained by means of variational geometric partitioning [CSAD04]. According to this framework, homogeneous regions of the surface are located by selecting points in a random way and adopting region growing strategies. At the end of the process, the lines separating the different regions characterize ridges and valleys on the model, i.e. the features searched for.

In our work, for the automatic extraction of 3D features from architectural models, an algorithm very close to the one by Cohen-Steiner, Alliez and Desbrun [CSAD04] has been implemented.

The second group of methods refers the fitting to the data of high level geometric primitives. In the scientific literature methods exist for the least-squares fitting of spheres, cylinders, cones, and tori to 3D point data, and their application within a segmentation framework [MLM01, Bes88]. The main application areas of this techniques are reverse engineering of solid models from depth-maps and automated 3D inspection where reliable extraction of these surfaces is essential. We have applied these techniques to the architectural field by using specific geometric primitives and by defining a framework which is easily extendible to other geometric primitives.

The project we are developing aims to supply the user with efficient tools for the extraction of information and measurements from dense 3D digital models. Most of the techniques and solutions presented in this paper have already been published. Commercial application packages as well present measurement and feature extraction capabilities; a not exhaustive list includes RapidForm by Inus [†], 3D Reshaper by TechnoDigit [‡], Applications 3D [†].

Our solution present a complete set of tools to measure, to visualize and analyze, to draw 3D primitives exploiting the underlying model, to extract features and to fit high level primitives. All these tools are integrated in a common, user friendly interface. Moreover, thanks to the used ad hoc geometric representation data structure [CGG*05], the user does not have to be concerned about the handling nor the geometric simplification of the digital models.

2. Our Framework

As mentioned in the introduction, the tool we provide in out application can be grouped into three groups:

Measurements and Utilities A set of tools to interactively visualize the model and to navigate the regions of interest, to carry out linear measurements, to locate sections of the models by means of parallel planes, to produce snapshots of the digital model (for large format printers) not limited to the screen resolution;

3D interactive design The user can *draw on the model*. The primitives outlined on the screen undergo an inverse projection transformation, from the screen back to the model; they lean against the model becoming real 3D primitives. These primitives, both lines or regions are measurable and exportable;

Reverse engineering Both the feature extraction modalities have been implemented: the automatic location of features as edges separating two homogeneous regions and the fitting to the data of high level geometric primitives.

In this Section we will give details about the most important aspects of the implemented software tools.

Before proceeding further it has to be underlined that a spatial indexing data structure is used to carry out most of the operations performed on the model. A 3D indexing data structure is initialized at the mesh loading time. Each element of the structure, a regular 3D grid, holds references to faces of the mesh standing on it. The data structure allows to speed up the location of the primitives to be used in the different geometric operations.

2.1. Measurements and Utilities

Among the facilities the program puts at user's disposal sections and high resolution rendering have to be pointed out. In order to compute the **sections** between a plane and the 3D model the user can directly use the analytic expression of the plane normal and a distance from the origin of the coordinate axes or she/he can, more simply, select three points on the model: the sections can be obtained for equidistant parallel planes as well. The vectorial *edgemesh* data structures, corresponding to the sections, can be exported in multiple common formats. Some examples of multiple parallel sections are shown in Figure 1.

A common and diffuse need in architecture is the generation of **snapshots of the scene** (prospects, views) to be printed on large format output devices (plotters). Our system includes the generation of high resolution snapshots. The resolution is not limited to the screen resolution because the image plane defined by the user is automatically subdivided into parts by means of a regular grid. For each grid element, the scene is rendered at the maximum resolution available for the frame buffer in use. The final image is built by simply collocating the partial views in the right position. Moreover, the user can easily select the appropriate projection type. In Figure 2 views of a 3D model in perspective, orthographic, cavalieri, and isometric projection are shown.

2.2. 3D interactive design

An important aspect in the process of recovering 3D information out of a huge 3D digital model of a building is rep-

---

[†] http://www.rapidform.com/62

[‡] http://www.technodigit.com/en1/En_software.htm

[†] http://www.applications3d.com/service_reverseeng.html

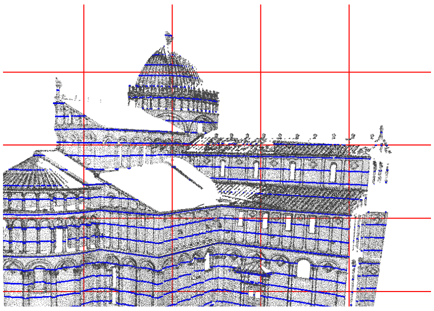

Figure 1: *Multiple sections between horizontal parallel planes and the model of the Dome of Pisa*

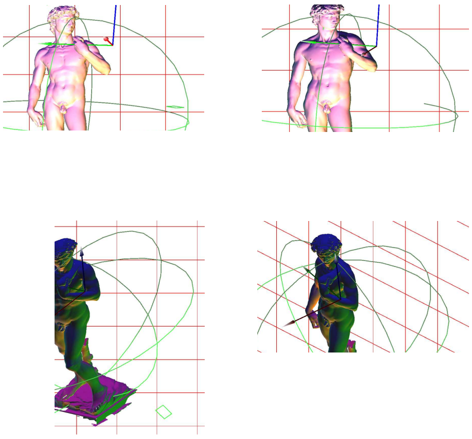

Figure 2: *High resolution output of the perspective (top left), orthographic (top right), cavalieri (bottom left), and isometric (bottom right) view of the David.*

resented by the possibility *to design* geometric primitives by exploiting the underlying 3D model.

In our system we support **the design of polygonal lines and close surfaces**. The importance of this tool is really high. The simple drawing of lines on the model permits the user to design directly in 3D starting from the acquired state of the structure. Locating regions in 3D allows the measurement of particular regions of the building which are not necessarily delimited by edges of the model; a typical example are the parts of the external surface to be cleaned or restored.

In the case of lines the user selects on the model the main points. By means a process of back projection constrained to the geometric primitives of the model itself the drawn polygonal line lays down to the model and follows its shape (see Figure 3).

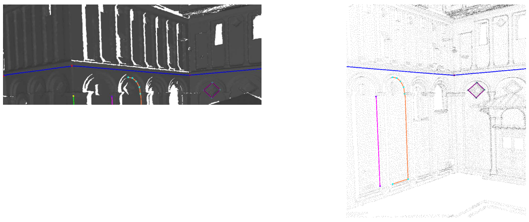

Figure 3: *Designing 3D polygonal lines on the digital model*

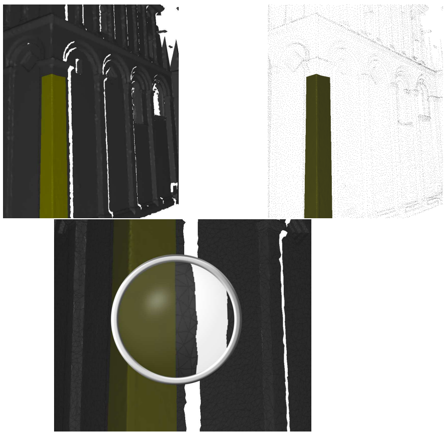

Figure 4: *Designing 3D closed regions on the digital model*

In a similar intuitive and assisted manner the user can select regions (see Figure 3). While the vertices of the region are located, the projection of the corresponding polygonal line is interactively updated in order to help the user in refining her/his selection.

The basic algorithm for the creation of the region [**?**] simulates a particle physical system; as a very close-fitting tissue, the user defined region approximates the part of the model the user is interested in. The boundary of the region is independent from the geometric primitives of the model (i.e. the edges of the triangles of the mesh are not necessarily the edges of the selected region). The only constraint for the correct behaviour of the physical system is that the polygonal region be homeomorphic to a disc.

The adopted solution presents a property which turns out to be really positive when dealing with range scanning data: all the holes belonging to the selected area are closed by fitting tissue.

2.3. Reverse Engineering

The **automatic recovering of edges** is an important component in the reverse engineering process. The recognized features can be directly exported in a CAD application software. The method we implemented in our system adopts a region growing approach and consists of two main modules:

Figure 5: *Automatic recovering of edges in a 3D digital model*

region growing which identifies uniform regions of the model; and

edge detection which attends to the location of edges as the result of the intersection between the interpolating planes of two adjacent regions.

The first module subdivides the model into *uniform* regions; each region R is characterized by a plane Γ_R and it contains one or more adjacent faces of the model. Each face of R belongs to Γ within a user defined ε tolerance.

At the beginning, the algorithm randomly selects some seed regions formed by single faces; for each region R, Γ_R is the plane the face belongs to.

The region growing approach proceeds by visiting the 3D model in a topological way and binding new faces to the appropriate regions. Obviously, each new face has to be adjacent to the region and respect the tolerance. Moreover, the algorithm is able to merge adjacent regions; again, the constraint is that the characteristic planes of the regions to be merged be ε tolerant.

The next step of the algorithm is the edge detection. In general terms, the characteristic planes R_1 and R_2 of two adjacent regions are intersected to locate the line L_{12} which the edge belongs to. Practically, the line needs to be limited in order to pick out the edge(s) on it. To do so, the extents of the faces of the regions R_1 are first projected onto the line. Each projected extent generates an interval I_1^i on L_{12}; the union of all the intervals $\bigcup_i I_1^i$ represents the contribution of R_1 to the searched edge(s). A similar job on R_2 brings to the set of intervals $\bigcup_j I_2^j$. The intersection of the contributing intervals $(\bigcup_i I_1^i)\bigcap(\bigcup_j I_2^j)$ identifies the edge(s) we are looking for. This approach ensure the removal of *false positive* in the edge recovering algorithm. It has also to be underlined that the projection of the extents of the faces of the regions onto the line is limited to the faces close to the line itself. This limitation avoids the interference with far faces not really contributing to the edge formation.

In order to exploit the architectural nature of our data, we endowed our feature extraction algorithm with a simple query management system. The user is able to define the edges she/he is interested in by simply expressing angles between adjacent regions or tolerances. An example of automatic edge detection result is shown in Figure 5.

Another important aspect of the reverse engineering process in our application is **the automatic fitting of high level primitives** to the 3D data. The least squares fitting of primitives is not a new approach. The relevant aspect of our solution is the ease for the user to extend the capability of the application and to implement new primitives fitting tools.

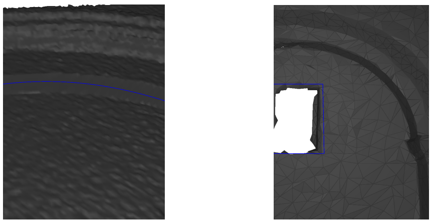

Figure 6: *Automatic fitting of high level geometric primitives. Fitting a roman arch (left) and a window (right).*

Figure 7: *The user-friendly interface of our application*

At present, the system is able to detect roman arches (of the largest class of the arches) and generic windows (an example in Figure 6); however, the basic organization in plug-ins makes it easy to write the new simple rules for the identification of other primitives belonging to existing classes or other classes and to expand the availability of fitting tools.

3. Results

We have sketched the tools we provide in our application for recovering 3D information from an architectural model. The best way to present the results of our work is probably to show images of the user-friendly interface and of the returned information.

Figures 7 and 8 show the application GUI and some of the edges automatically recovered by the system, respectively.

From the point of view of the timings, it has to be noted that the edge detection algorithm is really time consuming. It took **12.5 minutes** to recover the information shown in Figure 8. On the other hand, this phase is completely automatic, it is performed once-for-all and it can be carried out in an unattended manner.

4. Future work

The future development of our system for the recovering of useful 3D information from range scanning data is oriented towards two main direction: (a) the automatic fitting of new families of geometric primitives and (b) the *regularization* of the data for a CAD environment.

Because we are mainly interested in buildings with some cultural or artistic relevance, we are going to expand the families or classes of objects to be automatically recognized: columns, capitals, roses, and so on. This implies the definition of new plug-ins. In a second step, the members of

Figure 8: *Another application of the edge detection tool*

these families will be progressively populated. No more just roman arches but also full-centre arches, segmental arches, pointed arches, etc.

In a similar way we will have double lancet windows or triple lancet windows in the windows family.

The second aspect we intend to deal with in the next future is the so called *regularization* of the recovered information for an effective use in a CAD system. By regularization of the data we mean the possibility for the user to fix *architectural* requirements. Simple needs as, for example, walls of a building to be vertical or the windows on a wall to be at same height with respect the floor are not easily obtained by processing range scanning data. The tools we are going to design will help the user of our application in this direction.

Acknowledgments

This work was mainly supported by the CONTURA Project, a curiosity driven internal project of ISTI-CNR, Pisa (I).

References

[Bes88] BESL P. J.: *Surfaces in range image understanding*. Springer-Verlag New York, Inc., New York, NY, USA, 1988.

[BSZF99] BAILLARD C., SCHMID C., ZISSERMAN A., FITZGIBBON A.: Automatic line matching and 3d reconstruction of buildings from multiple views. In *ISPRS Conference on Automatic Extraction of GIS Objects from Digital Imagery* (Munich, 1999), pp. 69–80.

[CCG*03] CALLIERI M., CIGNONI P., GANOVELLI F., MONTANI C., PINGI P., SCOPIGNO R.: VCLab's tools for 3D range data processing. In *VAST 2003* (Bighton, UK, Nov. 5-7 2003), D. Arnold A. C., Niccolucci F., (Eds.), Eurographics, pp. 13–22.

[CGG*04] CIGNONI P., GANOVELLI F., GOBBETTI E., MARTON F., PONCHIO F., SCOPIGNO R.: Adaptive tetrapuzzles: Efficient out-of-core construction and visualization of gigantic multiresolution polygonal models. *ACM Trans. on Graphics (SIGGRAPH 2004) 23*, 3 (2004), 796–803.

[CGG*05] CIGNONI P., GANOVELLI F., GOBBETTI E., MARTON F., PONCHIO F., SCOPIGNO R.: Batched multi triangulation. In *IEEE Visualization Conf. Proc.* (Minneapolis, MI, USA, 2005), pp. 1–8.

[CM02] COLOMBO L., MARANA B.: 3d building models using laser scanning. *GIM - Geomatics Info Magazine 16*, 5 (2002), 32–35.

[CMS97] CIGNONI P., MONTANI C., SCOPIGNO R.: *A Comparison of Mesh Simplification Algorithms*. Tech. Rep. 97-08, Istituto CNUCE – C.N.R., Pisa, Italy, June 1997.

[CSAD04] COHEN-STEINER D., ALLIEZ P., DESBRUN M.: Variational shape approximation. *ACM Trans. Graph. 23*, 3 (2004), 905–914.

[LT99] LINDSTROM P., TURK G.: Evaluation of memoryless simplification. *IEEE Transactions on Visualization and Computer Graphics 5*, 2 (1999), 98–115.

[Lue01] LUEBKE D. P.: A developer's survey of polygonal simplification algorithms. *IEEE Comput. Graph. Appl. 21*, 3 (2001), 24–35.

[MLM01] MARSHALL D., LUKACS G., MARTIN R.: Robust segmentation of primitives from range data in the presence of geometric degeneracy. *IEEE Trans. Pattern Anal. Mach. Intell. 23*, 3 (2001), 304–314.

[OBS04] OHTAKE Y., BELYAEV A., SEIDEL H.-P.: Ridge-valley lines on meshes via implicit surface fitting. *ACM Trans. Graph. 23*, 3 (2004), 609–612.

[SB03] SCHINDLER K., BAUER J.: A model-based method for building reconstruction. In *First IEEE International Workshop on Higher-Level Knowledge in 3D Modeling and Motion Analysis* (2003), pp. 74–82.

[WB01] WATANABE K., BELYAEV A. G.: Detection of salient curvature features on polygonal surfaces. *Comput. Graph. Forum 20*, 3 (2001).