Processing of Terrestrial Laser Scanning Point Cloud Data for Computational Modelling

Short Running Title: Terrestrial LiDAR to Solid Models

*Debra F. Laefer[(1)], Linh Truong Hong[(2)], and Michael Fitzgerald[(3)]

[(1)] Corresponding Author, Urban Modelling Group (UMG), School of Architecture, Landscape, and Civil Engineering (SALCE), University College Dublin (UCD), Newstead G25, Dublin, Ireland 4 (353) 1-716-3226; (353) 1 -716-3297 (fax); Email: debra.laefer@ucd.ie
[(2)] Doctoral candidate, UMG, SALCE, UCD, Newstead G67, Dublin, Ireland 4 (353) 1-716-3232; (353) 1 -716-3297 (fax); Email: linh.truong-hong@ucdconnect.ie
[(3)] Former postgraduate student, UMG, SALCE, UCD, Newstead G67, Dublin, Ireland 4 (353) 1-716-3232; (353) 1 -716-3297 (fax); Email: michael.fitzgerald1@ucdconnect.ie

Abstract: With the rapidly increasing availability of laser scanning data and the growing pressure to use it as the basis for computational models, there has been heighten interest in quickly, cost-effectively, and accurately processing the resulting point cloud data sets so that they are compatible for importation into computational models. This paper presents traditional strategies for solid model generation and examines in detail innovations and continuing limitations of recent patents, newly published research, and some currently available commercial programs for the transformation of laser scanning point cloud data into solid models appropriate for finite element method meshing and various meshing strategies particularly well-suited for point cloud data.

Keywords: Terrestrial laser scanning, light detection and ranging (LiDAR), solid model generation, façade reconstruction, surface reconstruction, finite element method.

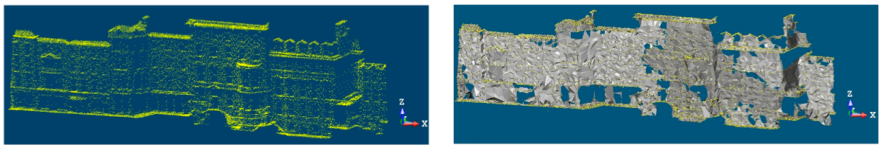
1. INTRODUCTION

Laser scanning [also known as Light Detection and Ranging (LiDAR)] has been applied to a wide variety of industries: forestry management [1], floodplain mapping [2], landslide monitoring [3] hazard identification [4] and more recently condition assessment [4-7]. Laser scanning works by laser triangulation, phase shift, or time of fight [8] and can be used from an aerial or terrestrial unit. Aerial laser scanning (ALS) was introduced commercially for digital terrain/elevation model (DTM/DEM) generation in the early 1970s, with the first generation of terrestrial laser scanning (TLS) equipment launched in 1998 [9]. Both approaches have geometric limitations depending upon the scanner's location, but generally the aerial has been used to collect lower density data sets for large areas such as in topographic modelling, while the terrestrial has been more often applied to capturing short to medium range objects (up to 250m) for which higher accuracy is needed (e.g. bridge documentation). TLS can achieve millimetre level accuracy but requires significant time for data collection [10], as opposed to aerial laser scanning (ALS) for which many kilometres of ground can be documented in an hour but with only sub-metre accuracy [11]. However, the introduction of truck-mounted TLS units and improved ALS hardware combined with innovations in flightpath planning [12] are making the distinctions between TLS and ALS less clear. Irrespective of the data collection approach, documentation of even relatively small areas can generate data sets of hundreds of millions of points. Thus, processing them in a manner that they are useful beyond topographic documentation is challenging. The challenges relate not only to the size of the data sets but in the fact that the individual data points are captured

without any inherent affiliation to an object or group of objects and that they themselves contain no geometric information. The LiDAR point clouds consist only of individual points with x-, y-, z-coordinates, affiliated red-green-blue values, and intensity values. This format is a major obstacle for the engineering usage of the data, especially given that the associated firmware for ALS and TLS systems are devised primarily for data viewing [3, 8].

Recent efforts in micro-climate modelling [14] and tunnelling subsidence prediction [15] indicate an emerging trend of using laser scanning data (both terrestrial and aerial) as the basis for not only documenting existing structures but also for populating complex computational models to achieve greater architectural detailing. As pressures increase for better environmental modelling and for more subsurface transportation, the need for accurate and economically generated computational models is likely to grow and with it increased efforts to use laser scanning data as its basis.

Although there are commercial solutions for the transformation of laser scanning point cloud data into solid models as the basis for computational modelling, many of the available processes have significant limitations including the following: (1) restriction to only surface mesh creation; (2) reliance on pre-established geometries, which are regular and planar and may not reflect an existing building's geometry; (3) dependence upon a computer-aided drawing (CAD) based program, which relies upon straight lines and uninterrupted planes resulting in unintentional "correcting" of the condition of any displaced/distorted structure; such auto-correcting prevents mesh convergence, without which a computational mesh is unusable; and (4) an inability to overcome sparse and missing data (Fig. **1**). Usage of an intermediary program may also result in loss of accuracy and/or a lengthy transformation process coupled with significant financial commitments to overcome convergence problems as previously noted [16].



a) Original point cloud data b) TIN conversion

Fig. **1**. Illustration of the difficulty in addressing sparse and missing data on a series of building facades in a traditional solid model generation approach.

To better understand how the current state of technology is developing and the contributions and limitations of recent advances on surface reconstruction for computational modelling from laser scanning data, this paper gives an abbreviated history of solid modelling and explains the dominant techniques used to generate a solid model over the last 40 years. A brief introduction of laser scanning applications in civil engineering and common solid modelling techniques used in computer-aided drawing (CAD) or computer aided manufacturing (CAM) is then provided. The next section outlines basic common methods of surface reconstruction from laser scanning data in computer vision and computer graphics and a sampling of commercial software to create solid models from data collected from existing objects' surfaces. Subsequently, trends and possible future developments are discussed.

2. SOLID MODEL GENERATION

A solid model can be considered as the geometric precursor to a computational mesh, such as is needed for Finite Element Method (FEM) modelling. In the 1970s, significant research was undertaken in solid model architecture. Two dominant methods that emerged were (i) Constructive Solid Geometry (CSG), which construct objects by Boolean operations, store CSG tree, and compute boundary representation; and (ii) Boundary Representation (BRep), which stores objects boundaries regardless of construction method or spatial subdivision, and decomposes a solid into a cell with a simple topological and geometric structure, such as the octree [17, 18, 19]. Brief applications of these are discussed in the following sub-sections.

2.1 Constructive Solid Geometry

In Constructive Solid Geometry (CSG), a solid structure is constructed from a series of simple primitive solids [e.g. box, sphere, cylinders or torus (Fig. **2**)] and Boolean operations (i.e. algebraic expressions such as union, intersection, and difference). Both the surface and the interior are implicitly defined [17, 19]. A special data construction is called a CSG-tree, where the basic solid shapes are leaves and Boolean operations or non-singular transformations are stored as interior nodes. Fig. (**3**) illustrates the CSG process via a simple example. Firstly, the individual basic shapes of the spherical tank with two cylindrical pipes are modelled as shown in Fig. (**3a**). These basic objects including a sphere and cylinder were subsequently combined by a series of Boolean expressions (rotation and union) to achieve the final object in Fig. (**3b**) [19]. Advantages of this process are that the solid model produced is continuous and watertight, containing no openings, which reduces the problem of discontinuities in mesh convergence. One of the major drawbacks of the CSG is that there is not a unique CSG tree representing the specific object [20]. Consequently, comparing two identical solid objects is a difficult task due to the large amount of possible trees that could represent that object. Goldman [19] provides an extensive description of these techniques, along with their relative advantages and disadvantages.

Fig.**2**. Primitives for a solid modelling systems: a box, a sphere, a cylinder, a cone, and a torus [19]

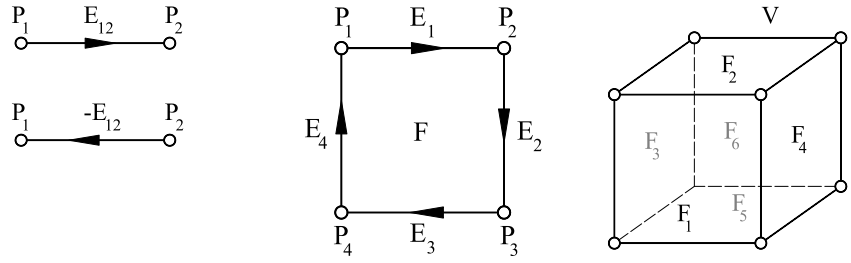
a) A spherical tank with two cylindrical pipes

b) A CSG tree with primitive solids and Boolean operations

Fig. **3**. Constructive Solid Geometry [19]

2.2 Boundary Representation

In Boundary Representation (BRep), the solid surface is described as a blanket of faces, edges, and vertices (Hoffman and Rossignac, 1996) [17, 19]. In this approach, relationships between entities in terms of topology are recorded. The geometry consists of vertices, curves, and surfaces along with numerical data describing position, size, and orientation, while the topology models adjacency and connectivity consisting of vertices, edges, and faces, along with pointers storing connectivity information (Fig. **4**) [17,18,19]. Some commercial packages use this technique to describe the solid model (e.g. ANSYS program [21]). These models are closed and contain no breaks. According to Bruenet and Navazo [22], this approach is highly effective for achieving visual and graphical outputs and calculating volumetric properties, but Boolean operations between solids or the entities involved can become quite complex. Velayutham [20] noted that in this approach, a large data structure is required even for simple boundary models due to the fact that the data are listed in a face-by-face manner. As a result, each edge is part of two faces, and thus, a significant amount of repetition of information is required, which is a major disadvantage in terms of storage efficiency.



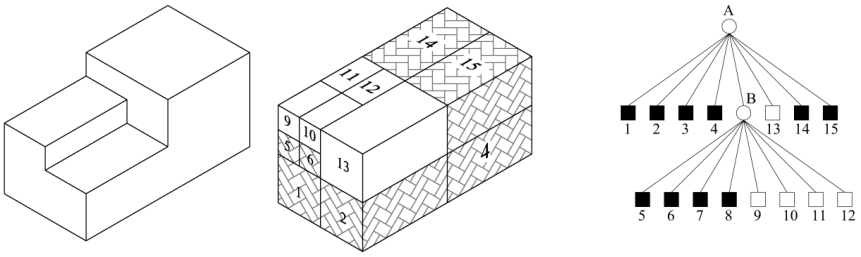
a) Vertices (P) and edges (E) b) Face (F) c) Volume (V)

Fig.**4**. Label and organization of vertices, edges and faces of the cubic by B-Representation

2.3 Octree Representation

While the above described techniques are very powerful as standalone methods for representing individual objects, they are poor choices for storing large datasets, because the number of objects and operators required for full storage is extremely memory intensive. The octree provides an alternative.

The octree is a data indexing structure, which was first suggested by Jackins and Tanimoto in 1980 [23] and further developed by Meagher [24]. In that approach, a cube is placed around a three-dimensional (3D) space called the bounding box and recursively subdivided into eight equal smaller cubes (Fig. **5a, b**) until the octree representation reaches terminated conditions, such as a depth threshold. Each resulting cube (often referred to as a voxel) constitutes an object represented by the nodes in a tree structure (Fig. **5c**). The root is assumed to be at the top of the node structure, while the other nodes are below the root [24]. Each voxel in the tree has a property to describe its status. Normally, their properties are either EMPTY or PARTIAL or FULL indicating that a voxel is either fully empty of the object, partially filled with the object, or completely occupied by the object, respectively [24]. Figure 4c shows a partially filled voxel recursively sub-divided into 8 child voxels (Fig. **5c**).



a) Example 3D object b) Octree block decomposition c) Tree structure representation

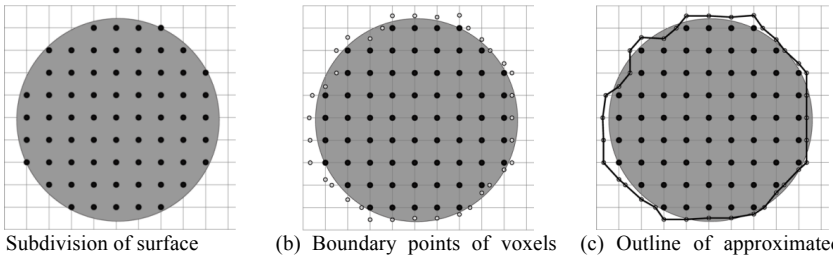
Figure **5**. Octree representation [adapted from 25]

Meagher [24] noted that one of the major advantages of this storage system is that it eliminates the need to rely on pre-sorted data prior to editing existing objects or creating new ones. Because of its hierarchical nature, algorithms can operate at a level appropriate to the task, thereby eliminating the execution of processes on out-of-scope data. Weyrich et al. [26] comment on the octree's ability to quickly locate a point in space, by exploiting binary locators to address octree cells, which is useful when used in tandem with efficient neighbourhood queries, such as the k-nearest neighbour algorithm as demonstrated by Gopi and Krishnan [27], as will be further discussed later in this paper. Although octree representation can construct the object with less accuracy than the two aforementioned techniques [19], the octree has become popular for modelling and visualizing objects in animations and other applications, such as surface reconstruction from digitized data, because of its ability to model the complex shape of an arbitrary topology [28]. Some work on surface reconstruction using octree representations from laser scanning data are discussed below.

3. SURFACE RECONSTRUCTION

3.1 Surface reconstruction algorithms

In the area of reconstructing real models using 3D laser scanning data, many approaches have been proposed in the computer vision and computer graphics communities over last two decades. In this section some common methods are presented that may be relevant to reconstruct facade buildings from the LiDAR data; for other basic methods readers can refer to Mencl and Muller [29]. Lorensen and Cline [30] developed a powerful and extensively adopted technique to determine a surface model from 3D data entitled the Marching Cubes algorithm for medical scan data such as that generated during computed tomography (CT), Magnetic resonance (MR), and single photon emission computed tomography (SPECT); such data can be considered analogous to TLS data. The algorithm is a two-step process: (1) locating the surface corresponding to a user-specified value by cutting through a cube and determining whether the cubes vertex lies above or below the scan's surface and (2) calculating the normals to the surface at each vertex of the triangle. There are 256 (2^8) cases in which a surface can intersect the cube, as there are eight vertices in each cube and two states of surface: inside and outside. In case of two different symmetries that are the topology of triangulated surface and rotational symmetry, there are really only 14 configurations instead of 256 possibilities. The process occurs by subdividing the space into a series of small cubes (or voxels) [Fig. (**6a**)]. The algorithm starts at a point and tests each cube. Should a cube contain an element of the surface geometry, it then replaces the empty cube with a series of triangles around the point cloud based on vertices on the boundary of the cell in close proximity to the surface. These are joined together to form polygons [Fig. (**6b**)]. The algorithm then moves to immediately adjacent cubes to find neighbouring vertices to produce an approximated polygonal mesh of the surface [Fig. (**6c**)].



a) Subdivision of surface

(b) Boundary points of voxels (white circles) chosen as vertices for triangles to approximate the surface

(c) Outline of approximated voxels polygonal mesh produced from marching cubes algorithm

Fig. (6). 2D graphical version of marching cubes algorithm [adapted from 30]

Similarly, Olson [31] proposed using a marching cubes algorithm to reconstruct surfaces from sample points representing an object. In this patent, point cloud data are binned into an n-dimensional array of elements for which a binary value is associated with each cell. The dilation algorithm calculates and assesses a genus of the bin's contents. Subsequently, an erosion algorithm is applied to the dilated binary representation of the points to output a segmented volume. This reduces the time and overhead required to reconstruct complex and non-convex surfaces from a point cloud.

Although the Marching Cubes algorithm has been shown to work effectively on real test cases including on arbitrary topologies, the resulting meshes suffer from nearly singular triangles and poor approximation of shape features [32]. To improve the geometry approximation and quality of the mesh in the early phase of surface reconstruction, Azernikov and Fischer [32] proposed a new approach based on connectivity graph approximation from the Hierarchical Space Decomposition Model (HSDM) and facet reconstruction. Unlike methodologies used by Hoppe et al. [33], Curless and Levoy [34], and Bernandini et al. [35] (as well be subsequently described), a distance function from a set of range images to approximating a surface is not defined. Instead, the zero contour is extracted via the Marching Cubes algorithm. The algorithm has two steps composed of mesh reconstruction from a point cloud without additional information and subsequently extended mesh by incorporating normal directions (Fig. **7**). In this, three phases exist for each stage: (1) HSDM construction from raw point cloud data, (2) surface extraction, and (3) feature classification. The reconstructed surface is approximated by a mesh composed bi-linear facets, which has a good aspect ratio and is quadrilateral almost everywhere. The algorithm is very flexible and suitable for processing large-scale 3D data. To demonstrate this four models were constructed including a mechanical part, car, toy airplane, and an oil pump. The Hausdorff distance was used for error estimation, with values from 1-2 % (see Table **1** for sample results) [32].

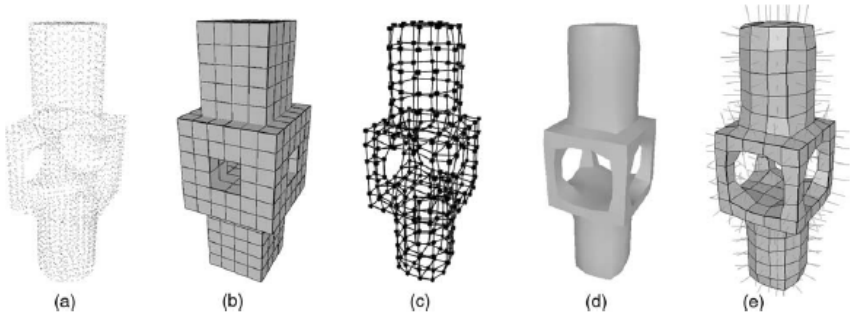


Fig. **7**. Reconstruction process phases: (a) point cloud, (b) HSDM, (c) connectivity graph, (d) facet reconstruction and (e) anisotropic smoothing [37].

Table **1.** Results from HSDM methodology [32]

Object	A number of sample points	Octree depth	Error (%)	Execution time (s)
Mechanical art	4,102	4	2.0	0.1
Car	20,623	5	1.0	0.2
Toy airplane	117,152	6	1.1	3.9
Oil pump	166,087	6	1.4	6.0

Hoppe et al. [33] presented an algorithm for reconstructing a 3D surface from a set of unorganized points scattered on or near surface irrespective of the presence of a boundary. The approach was based on the idea of determining the zero set of an estimated signed distance function (level set). The bound-

ing box of the dataset was subdivided into a regular, voxelized grid and an estimate of the signed geometric distance to the unknown surface based on a normal and tangent vectors though a covariance matrix of neighbouring sample points. Points on the surface have distance 0, while points have positive and negative distance, if they are outside and inside of the surface, respectively. The Marching cubes algorithm [30] was employed to separate the surface patches based on the sign distance at the vertices of voxels. The resulting triangular mesh separates the positive and negative distances. Gross and Thoennessen [36] used the assumption of a normal vector of a tangent plane for each data point proposed by Hoppe et al. [33] to determine features of sample points based on analyzing a covariance matrix including all neighbouring points defined by a sphere. By comparing a new feature of sample points with the analytical results of typical point configurations provided a discriminating feature from which to extract points, which may belong to border lines of man-made objects. Similarly, Schuster [37] used tensor voting (determined from a covariance matrix of a neighbourhood of given points) to classify sample points into points, surfaces, and lines separately from within TLS data for reconstructing building facades.

Curless and Levoy [34] defined a volumetric function based on an average of the signed distance from each range image, in which the volumetric function at points on a uniform 3D grid can extract the zero crossings. This approach may fail to detect features smaller than the grid spacing and also requires a significant amount of space, as well as execution time. To overcome this drawback, the run-length encoding technique was employed. Pulli et al. [38] proposed a solution creating meshes from multiple range maps, where a triangular mesh can be extracted from an octree representation of an object. Cubical volumes were classified as inside, boundary, and outside according to their respective location to the sensor and the range data. The boundary cube was recursively subdivided into eight smaller cubes, which were added to the octree as children of the current cube. The volume representation has to recur up to the finest level possible to accurately determine the boundary. The method also automatically fills holes due to gaps in input data. Dalmasso and Nerino [39] improved reconstructed surface accuracy by using a volumetric approach to describe objects at different spatial scales. The estimated surface was then obtained as the zero level of the final global volumetric function based on surface voxels of the octree structure.

Similar to the previous work of Hoppe et al. [33] and Curless and Levoy [34], Osher and Thiyanaratnam [40] proposed surface reconstruction data processing based on a 3D rectangular grid as the zero level set of a function. In this work, the 3D grid may be rearranged into a two-dimensional (2D) grid where the data are compressed and stored in the form of a gradient. In order to recover the point cloud**,** the 3D grid is rebuilt from the 2D data and an interpolating algorithm on the implicit function is utilized to compute the points on the surface. The exact distances from the three-dimensional 3D grid to each point cloud in dataset were determined by the Eikonal equation, and those distances were used as fixed boundary values.

Edelsbrunner and Mucke [41] proposed a method to reconstruct surfaces from a given sample point based on Delaynay tetrahedrization and a Voronoi diagram. The tetrahedra, triangle, and edges of a

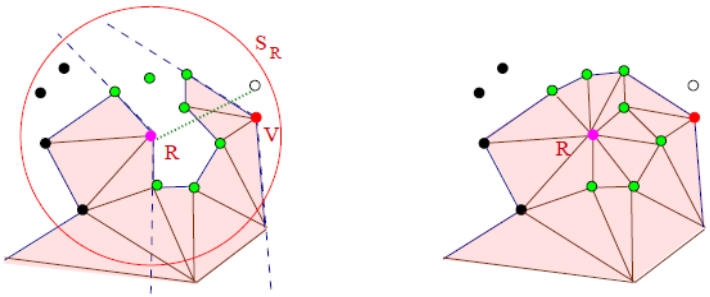
given Delaunay tetrahedrization were removed from the current mesh, if the minimum surrounding sphere with radius (called the α ball) cannot fit this tetrahedrization; a collection of points, edges, faces, and tetrahedra is called an α-shape. Subsequently, the triangle belongs to the desired surfaces, if there are two possible spheres of radius α through all three points of this triangle, and at least one of them does not contain any other point of the dataset. This approach was found to be sensitive to the α radius selected, and the surface may become fragmented. Also, based on properties of Delaunay triangulation meshes, in working on detecting windows/doors from the TLS data, Pu and Vosselman [41] and Boulaassal et al. [42] proposed that triangular sides of the triangles in the openings are greater than the ones of the triangles belonging to the solid walls, because the windows do not sample points available for triangularization. The approach can extract sample points from the same hole but often generates false positives, reporting openings that are in fact only missing data or data occlusions.

Boissonat [43] proposed a method to reconstruct a triangulation mesh of surfaces from a given set of points. Before the triangulation process starts, the neighbourhood of each point is defined: a first edge joins a given point and its nearest neighbour. In order to attach a new triangle to the edge, the approximate tangent plane was defined by applying a least squares method in the neighbourhood of the considered edge, and its neighbouring points were projected onto this plane. The new triangle was constructed by connecting a point to the considered edge, which sees this edge under the maximum angle. The triangulation meshes propagate until there are no free edges still available. Similar to the idea of an incremental surface, Pu and Vosselman [44] used a plane growing surface approach and human knowledge about building features (e.g. size, position, direction, and topology) to recognize potential building façade features from the TLS data. Similarly, Secord and Zakhor [45] used a region-growing algorithm to detect trees from ALS data. In this case, the segmentation grows, if adjacent data points have a point wise similarity above a threshold, which is computed from weighted features, such as height, colour, and normal vectors.

For identifying buildings from ALS data, Matei et al. [46] classified point cloud data into ground and non-ground points representing buildings and clutter, in which non-ground points were those that were higher than a predetermined threshold. The non-ground points were segmented into buildings and clutter. The processing segmentation included four steps: estimating local surfels of the non-ground points, grouping non-ground surfels into coarse regions by using a bottom up region growing technique, fitting points to the plane by employing an expectation maximization technique, and segmenting building regions from the planes. Contours of each building segment were extracted by using a ball-pivoting algorithm [47].

Gopi and Krishnan [27] developed a new projection-based, surface reconstruction algorithm from raw point clouds, which generates a triangular mesh based on a nearest-neighbour algorithm. The data structure is organized as a depth pixel array similar to the dexel structure as proposed by Hook [48], in which all data points are orthographically projected onto a 2D pixel array, and the points on the same pixel are sorted by their depth (z) values. The algorithm worked with two parameters: μ, which quantifies a locally uniform sampling, and α, which gives a maximum angle between two consecutive neigh-

bours of a point on a boundary of the surface. The algorithm starts at a reference point and finds all neighbouring triangles within a spherical radius centred at this point (Fig. **8a**). The radius of the influence sphere was adopted to be equal to μm. This means the distance ratio of the farthest and closest neighbours of a sample in the given sampling of the object are less than a constant value, and m is a minimum distance from the reference point to another point on the same layer. In an experimental setup, a set of candidate TLS points were extracted from neighbour points of the reference point, which points were on the boundary of incident triangle meshes and had not yet been chosen as reference points. The incident triangles were determined based on the normal deviation between two adjacent triangles. The remaining points were connected to the reference point to complete the triangle mesh of an object in space, but in case the angle between the two consecutive remaining points was greater than the predefined values, they were not connected to form a triangle (Fig. **8b**).



a) Visibility test around R (a magenta point) (the black points are behind R's boundary edges; the white points are occluded by other edges; the V is eliminated as R is behind its boundary edges; the green points are candidate points; two dash lines are consecutive boundaries)

b) Completed triangle mesh at R

Fig. **8.** Nearest neighbour algorithm [27]

This algorithm successfully reconstructed surfaces with point clouds of 10,000 to 100,000 points in 3 to 30 seconds (250 Mhz processor, R10000 SG1 Onyx2 with 16 GB of main memory). However, some drawbacks are the non-unique triangular meshes for a specific surface and bad approximations for sharp surface curvatures. Currently, many automated mesh generation methodologies are unable to accurately accommodate objects with complex typologies or curvilinear shapes. To help overcome this, Várady et al. [49] devised a shape reconstruction algorithm consisting of (1) hierarchical Morse complex segmentation, (2) feature skeleton construction, (3) computing region boundaries, (4) surface structure creation, and (5) surface fitting. Triangles are organized into primary regions or a separator set (i.e. connecting features). The final step can be conducted using either a quadrilateral or the more computationally intensive trimmed surface; their resulting accuracy was not reported.

In a patent by Hinks et al. [50] an octree was employed to help order the point cloud so that it can effectively be voxelized as the basis for solid models. In this work, the root voxel was recursively subdivided until the depth of the octree reached a predefined depth threshold, with all voxels sub-dived into eight child voxels at that level. The voxels were classified as either full or empty, depending upon

whether they contained a sample point. Subsequently, the full voxels were converted to a neutral file for generating solid model for computation by using boundary representation. The approach possesses the advantages of being applicable to both ALS and TLS, not necessitating further object identification, and not requiring supplemental information beyond the location of the scanner, but the approach requires user determination of the depth of the octree (for which no guidance is currently available), and the fact that boundary approximation is not inherently well defined.

Alternatively, a manual, user-guided reconstruction mechanism to correct imperfect scan data was proposed by Zhou et al. [51], in which surfaces were iteratively reconstructed from sample points using a two-step, octree-based approach. In the first step, an implicit function over the volume spanned by the octree nodes is computed using Poisson surface reconstruction. In the second step an adaptive marching cubes procedure extracts a watertight mesh as an iso-surface of the implicit function. The algorithm allows the user to draw strokes to reduce topological ambiguities in poorly sampled area and to specify the geometry of missing area of the surface, which is generated from another point cloud around the target area. Real time viewing of the image is facilitated by parallel surface reconstruction by using the tree construction mechanism to build data structures to produce the data for displaying the surface. In addition to these approaches there are commercial programmes developed explicitly for LiDAR point cloud transformation as described in section 3.2.

3.2 Sampling of commercial and industrial software packages

Commercial software for transforming point cloud data into surface representation or solid models can be described in two general categories: dependency upon a CAD-based intermediary or "add on" component of CAD platform and independent solutions [52]. There are many software packages that have been developed for surface reconstruction throughout last two decades. As an example, the capabilities of sampled software packages used for civil engineering are summarized in Table 2, along with their limitations related to data density. Output format files of these products are DXF, SAT, STEP, STL, IGES and others compatible with generic FEM packages.

3.2.1 KUBIT and Rapidform XOR programs

Many commercial programs process the dataset so that it is CAD compatible, and then the CAD image becomes the basis for any computational modelling. Examples are Kubit [53] and Rapidform XOR software [54]. Kubit converts point clouds into Autodesk AutoCAD Drawing (.dwg) compatible files [53] for importing to finite element commercial packages. The Kubit PointCloud offers many manipulation and editing tools to assist user(s) in reconstructing surfaces and 3D solids from the raw data with x-, y-, z-coordinates. One feature, entitled clash detection, is useful for comparing as-built conditions with a proposed CAD model inserted into the dataset (e.g. a new pipe needs to be installed and integrated into an existing pipe network as illustrated in Fig. **9).** Many of the in-built tools support the user defining an area of interest within the cloud, and then fading out irrelevant data, assigning a colour to a particular section, and automatically fitting lines and cylinders. Object edges and corners can be created from the intersection of defined planes. The user can also integrate high-resolution photographs taken by the scanner to overcome a region of low or missing data points. The program uses an octree to

store the positional data, along with associated semantics such as colour, intensity, and classification. The algorithm is applied to every point within a designated section from within which commands such as "Hide the points" or "Detect a corner" can be applied. Currently, Kubit is in the process of developing a polygon-fitting tool to be incorporated with Kubit PointCloud 6.0 [53]. Additionally, the recently released AutoCAD 2011 software has a native point cloud engine, which is capable of storing up to 2 billion points in 1 file. While a solid model can be produced from tracing around the point cloud data within AutoCAD using the KUBIT plug-in, a significant amount of manual manipulation is required to generate a solid model, as it must be a closed, watertight region in order for it to converge within a computational package.

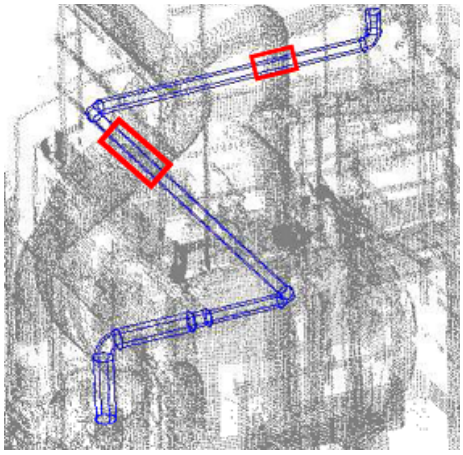


Fig. **9**. Proposed 3D Pipeline integrated into existing point cloud of pipe network [55]

Similarly, Rapidform XOR, another CAD reliant software, focuses on transforming terrestrial point clouds using a polygonal meshing algorithm based on a Non-uniform rational B-spline (NURBs) based approach [54]. It is capable of importing data from many 3D scanner systems and results of solid models can exported various format data for CAD/CAM [54]. In work assessing the structural safety of masonry vaults, Schueremans and Genechten [56] used Rapidform software after manual removing noise and redundant points from TLS data. There is an inherent trade-off with this in the manual post-processing time and accuracy. The grid size 10cm x 10cm incorporates 21 slices in the transverse and 39 slices in the longitudinal direction. The points were then extracted into a 2D format using a program called ObjectARX that fits a plane to a number of points and exports the co-ordinates in .txt format which can then be exported into the structural analysis package Calipous [57], which was developed specifically for structural analysis of masonry arches. It took one day to generate a 3D computational model of the sample vault.

This CAD-based approach has some distinct disadvantages as shown by Young *et al*. [58] in an analysis of studies conducted by Viceconti et al. [59], Schmitt et al. [60], and Wirtz et al. [61] in deriving medical computational models from raw medical data. The AutoCAD-based meshing algorithms were

time-intensive compared to automated mesh generation algorithms and that a satisfactory model could not be generated without a significant degree of manual post-processing. Often, the models produced contained a lot of unwanted voids or incomplete geometry due to missing data or noise in the original point cloud. The models produced were, therefore, not watertight and would not converge when extrapolated as a neutral file into a computational package. They also found that these algorithms run into considerable difficulty when trying to extract imaging data containing curves and complex typologies. They are often described in a piecewise linear manner fashion containing errors and gaps in geometry, thereby resulting in low accuracy geometric approximations. Lorenz [62] suggested a solution to this problem via integrating the mesh generation tool with a system-level design and simulation environment, thus enabling the direct generation of the Partial Differential Equation (PDE) solver input from this approach and a method for using a mesh generation tool. Secondary information about the nature and purpose of certain design components to the mesh generation tool are included to improve the automatic mesh generation and optimization. The mesh generation tool retrieves information from the individual components and connectors of a schematic of a MEMS device and produces a discrete element model suitable for numerical PDE analysis by the FEM and boundary element methods. The synergy between the system-level design and the PDE analyses allows the user to move between the two levels of abstraction. Unnecessary computation is avoided as a result of the direct link from the schematic to a mesh model aiding the user to choose an optimal mesh for PDE analysis.

3.2.2 ClearEdge3D's Edgewise and Geomagic Studio 12

As an alternative to using CAD-based intermediaries, there are commercial software packages that directly generate solid models from point cloud data, such as ClearEdge3D's Edgewise and Geomagic Studio 11. The ClearEdge3D's Edgewise [63] is marketed as capable of automatically deriving polygonal models. The algorithm behind the software is based on a pending patent believed to use straight-line fitting and merge based RANdom Sample Consesus (RANSAC) or similar techniques to that of Arefi et al. [64]. The program searches for boundary points on each co-planar surface from a registered point cloud. A best-fit rectilinear polygon is then fitted to these points on the boundary. Each region (i.e. a series of co-planar points) is described by a different colour for ease of identification. The 3D model produced is comprised of a series of observed planar surfaces fitted using least-squares regression [63]. Edgewise's preferred methodology is to process point clouds on a scan-by-scan basis rather than working with a merged scan. To achieve this, the program also requires that the scanner's location be known with respect to the object being scanned. This is an input parameter needed as part of the algorithm to derive the polygonal mesh [63]. Processing speeds depend on the quantity of data in the point cloud, complexity of the scene scanned, and computer processor speed. However, claimed processing time is typically 5-10 minutes, with 30 minutes for more complex scans on a mid-budget laptop. This is advertised to be 2-10 times faster than manual surface fitting, but no data has been published to verify this. The software has the advantage of being a stand-alone application.

A competing product is Geomagic Studio 12 [65], which is software capable of deriving a 3D polygonal mesh from raw point cloud data based on the principle of Delaunay triangulation. It has the capabilities of handling datasets of up 126 million points. The software can import an array of files

from all major terrestrial laser scanners and digitizers including the native ASCII format. A number of toolboxes are available including registration tools for merging numerous scans together and optimization tools for outlier removal and noise reduction. The point cloud density can be reduced to decrease processing time using the random, uniform, or curvature-based point sampling. Once the polygonal model is created, tools exist to modify, edit, and clean. Holes can be filled manually via a hole plugging functionality, and boundary edges can be sharpened.

3. 3 Comparative results

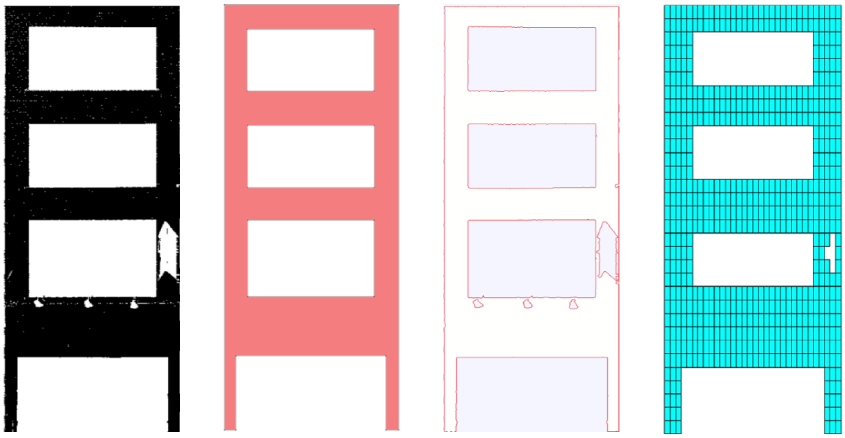
To provide a comparison of a sampling of manual and automatic techniques for surface reconstruction from laser scanning data, the TLS data of façades of three brick buildings in Dublin, Ireland were selected. These are short and medium sized buildings having rectangular and arch shaped windows. The datasets were acquired with a Trimble G200 [66] controlled by the RealWorks Survey (RWS) Advanced V6.3 software [67] installed on a PC linked to the scanner. In this test, the datasets were cleansed of all data involving internal walls/objects and of all vehicles or trees in front of the buildings. For each building, the solid models were reconstructed from two datasets: (1) the original point cloud (Fig. **10**a) and (2) one with a reduced sampling density, in the which distance between adjacent sample points was not less than 75mm (Fig. **11a, 12a**). All experimental tests were run on a Dell Precision Workstation T5400 with main system configurations following: Intel (R) Pentium (R) Xeon (8CPU) CPU speed 2GHz with 24 Gb RAM.

The façades were reconstructed from the above datasets using the commercial programmes Kubit and Geomagic and the procedure patented by Hinks et al. [49]. In Kubit, the solid models of these buildings were created based by the user manually identifying with the TLS data window openings and building boundaries based on a photograph of the building. For the other two approaches the solid models were reconstructed without any manual intervention, in accordance with their general workflow parameters [50, 65]. The resulting solid models from the three approaches for each of the three buildings are shown in Fig. **10 -12**.

In Kubit program, users create smooth boundaries of the façade and its openings by estimating the relative locations through comparison to photographs and/or user knowledge. Similarly, the problem of missing data points or unexpected holes due to occlusions is manually addressed (Fig. **10b**). However, this requires prior knowledge of the buildings and operator experience. For example, without a photograph of the building 2, rectangular windows were created by the operator for the 2nd and 3rd storeys, instead of the actual arched windows (Fig. **11**b). The automated approaches did equally poorly. There were also convergence problems with Geomagic, because the solid models contain some degenerate shapes (Fig. **10c, 11c, 12c**), such as the tip at atop of the 4th ground level door (Fig. **12**c). In addition, in the automated methods, the resulting solid model boundaries were rough and did not reflect the in-situ condition. Futhermore data holes were not filled automatically, such as three small holes below a large window in the first floor of the building 1 (Fig. **10**c). Similarly, using Hink et al. [50] to construct the solid models (Fig. **10d-12d**) some contained floating voxels [i.e. a voxel on atop of the 4th door (Fig. **12**c]. Furthermore, in this approach the solid models may contain fragment voxels if a high depth of

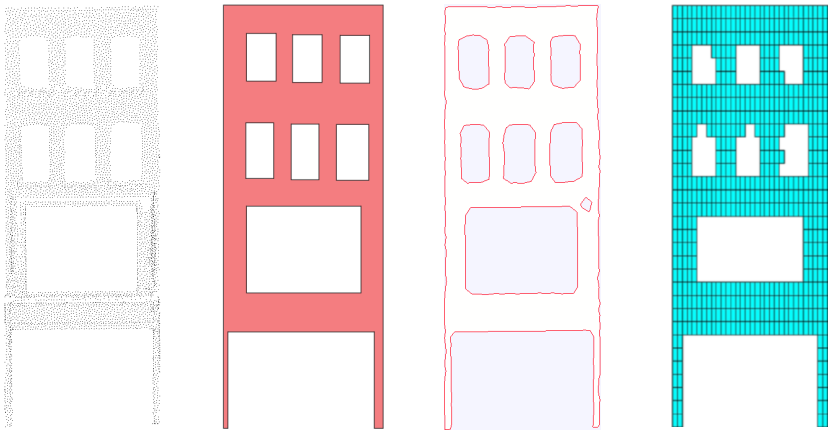
Debra Laefer 7/12/10 13:07
Comment: I would circle this in the image

Octree representation is defined, particularly size of voxels is less than distance between two adjacent sample points.



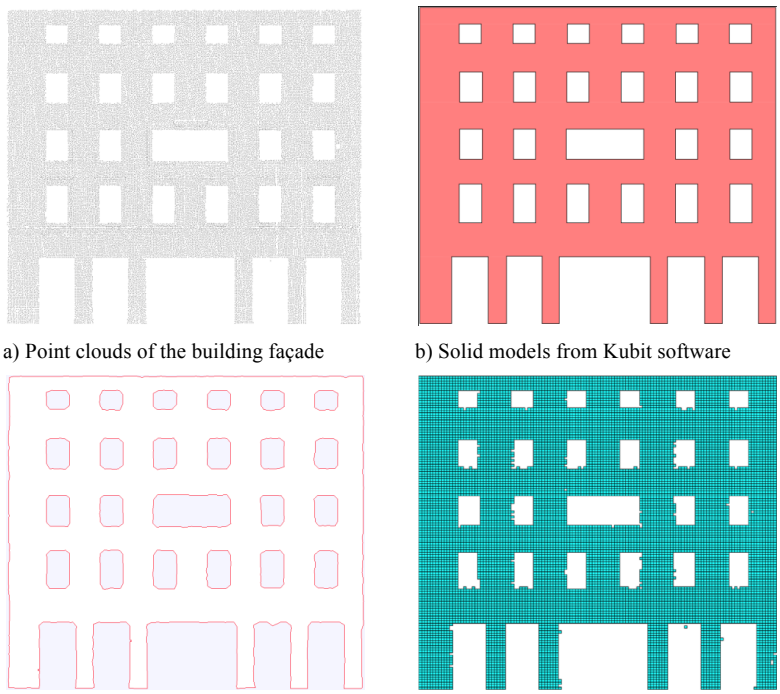
a) Point clouds of the building façade | b) Solid models from Kubit software | c) Solid models from Geomagic software | d) Octree representation using Hinks et al. [50]

Fig. **10**. A solid model of the building 1 (2 Anne St. South, Dublin, Ireland) reconstructing from original dataset by using different algorithms



a) Point clouds of the building façade | b) Solid models from Kubit software | c) Solid models from Geomagic software | d) Octree representation using Hinks et al. [50]

Fig. **11**. A solid model of the building 1 (5 Anne St. South, Dublin, Ireland) reconstructing from sampling density of the dataset by using different algorithms



a) Point clouds of the building façade

b) Solid models from Kubit software

c) Solid models from Geomagic software

d) Octree representation using Hinks et al. [50]

Fig. **12**. A solid model of the building 3 (2 Westmoreland St., Dublin, Ireland) reconstructing from sampling density of the dataset by using different algorithms

Output files compatible to generate solid models within finite element commercial packages, such as ANSYS [21], were considered (Table **3**). In term of storage, output files from Kubit require less storage space than ones of other programs. With the Hinks et al. [50] approach, the storage needs can be reduced significant by merging all voxels together before converting voxelization models to a suitable format file with the FE packages, such as a neutral file for ANSYS program [21], because the Hinks et al. [50] converts all full voxels into a neutral file. The solid models from both the Kubit and the Hinks et al. [50] approach were compatible with the FE commercial packages (e.g. ANSYS program) to generate meshes for computational modelling for these masonry structures. Geomagic's output was not compatible with an element appropriate for masonry modelling.

Table 3. Storage efficiency from different surface reconstruction approaches (MB)

Debra Laefer 7/12/10 12:55
Comment: I think we should only show the data that matches the images

Software	File format	2 Anne St. South		5 Anne St. South		2 Westmoreland St.	
		264,931 (pts)	4,643 (pts)	190,865 (pts)	5,366 (pts)	650,306 (pts)	35,468 (pts)
Kubit	SAT	0.035	0.035	0.061	0.064	0.208	0.193
Geomagic	IGES	266.0	4.5	189.0	5.23	598.0	34.5
Hinks et al. [50]*	ANF	9.9	9.7	9.7	9.7	193.9	192.0

*Octree depth is 6 in first two buildings and 8 in the last building

4. CURRENT & FUTURE DEVELOPMENTS

In addition to the work directly devised for pointcloud processing, there is a body of work from digital imaging that has been on a parallel track. The work is claimed to be significantly cheaper because of the hardware involved. An example is a recent patent by Xu and Yu [68] on the reconstruction of 3D surface models from body images using multiple two digital cameras. They propose a four-step approach: (1) data resampling; (2) initial mesh generation; (3) mesh simplification; and (4) mesh subdivision and optimization.

In not dissimilar work, Lubowiecka et al. [69] derived a FEM mesh of an existing king post timber truss in a historic building from a close range digital photogrammetry survey in Northern Spain. The post-processing of the data (i.e. a 3-D wire frame solid model) was subsequently edited in a CAD format prior to being extrapolated into a solid model within a commercial computational package package. Errors in the estimation of 3-D co-ordinates subsequent to capturing of data and importation into the CAD model process were 5-21 mm. Both 2D and 3D analyses were undertaken. The first model was a 2D FEM model consisting of single line beam elements. The second model was a more realistic 3D FEM representation depicting the irregularities and consisted of 3D solid elements. In work on reconstructing building facades, façade grammars were used to reconstruct repetitive components (e.g. doors and windows) from a range photo [70-72].

Another trend is the combination or amalgamation of multiple forms of remote sensing data as the basis for solid model generation. As an example, Quadling et al. [73] combined a structured light pattern digitizing method with photogrammetry to determine a 3D model of an object, which generates a more accurate model than photogrammetry alone. Similarly, based on both TLS data and range images existing techniques from image processing (i.e. Sobel filter, Canny extractor and Hough transformation) were employed to extract high order boundary of objects that may be lost when only the TLS data was used [74, 75].

Arguably, the three most persistently difficult areas for solid model generation from LiDAR data are as follows: (1) reconstruction of high order curves; (2) continued dependence on CAD-based platforms; and (3) reliance on a high-level of operator experience for data cleansing and feature recognition. The problems are generally greater for ALS data than TLS data because of its sparser data density. To date, most robust approaches for reverse engineering have been highly industry and/or product specific. A highly robust, generic algorithm for solid model generation for LiDAR data has yet to be devised.

5. CONCLUSIONS

There has been a strong interest in recent years to process point cloud data from laser scans to be suitable for use in computational models for environmental and urban concerns. While many commercial

software packages still rely on CAD-based intermediary programs, there is an increasing availability of approaches (commercial and research-oriented) that directly employ solid modelling, along with an optimization of meshing approaches to best suit the use of point cloud data in FEM meshes. This paper provides an overview of these efforts and documents the current capabilities and continuing limitations of available technologies and algorithms. Comparative analysis shows that without extensive manual intervention, overcoming missing and sparse data while simultaneously generating a geometrically accurate model is rarely achievable, even for highly regular, rectilinear structures.

LIST OF ABBREVIATIONS

Light Detection And Ranging: LiDAR
Terrestrial Laser Scanner: TLS
Airborne Laser Scanner: ALS
Computer-Aided Drawing: CAD
Finite Element Model: FEM
Constructive Solid Geometry: CSG
Boundary Representation: BRep
Computed Tomography: CT
Magnetic Resonance: MR
Single Photon Emission Computed Tomography: SPECT
Hierarchical Space Decomposition Model: HSDM
Non-Uniform Rational B-Spline: NURBs
Partial Differential Equation: PDE
RANdom Sample Consesus: RANSAC

ACKNOWLEDGMENTS

Support for this work was generously provided by Science Foundation Ireland, Grant 05/PICA/I830 GUILD: Generating Urban Infrastructures from LiDAR Data.

CONFLICT OF INTEREST

The authors have no conflicts of interest to declare.

REFERENCES

[1] Næsset E. Determination of Mean Tree Height of Forest Stands Using Airborne Laser Scanner Data. ISPRS J Photogrammetry & Remote Sensing 1997; 52(2): 49-56.
[2] Straatsma MW, Baptist MJ. Floodplain Roughness Parameterization Using Airborne Laser Scanning and Spectral Remote Sensing. Remote Sensing of Environment 2008; 112(3): 1062-80.
[3] Olsen MJ, Johnstone E, Driscoll N, Ashford SA, Kuester F. Terrestrial Laser Scanning of Extended Cliff Sections in Dynamic Environments: Parameter Analysis. J Surveying Eng 2009; 135(4): 161-9.
[4] Laefer DF, Pradhan A. Evacuation Route Selection Based on Tree-Based Hazards Using LiDAR and GIS. J Transportation Eng 2006, ASCE; 132(4): 312-20.
[5] Olsen MJ, Kuester F, Chang BJ, Hutchinson TC. Terrestrial Laser Scanning-Based Structural Damage Assessment. J Comp in Civ. Eng. 2010; 24 (3): 264-272.
[6] Laefer DF, Fitzgerald M, Maloney E, Coyne D, Lennon D, Morrish SW. LiDAR Lateral Image

Degradation And Its Implications For Condition Assessment And Structural Health Monitoring. Struct Eng Int 2009; 19 (2): 184-9.
[7] Laefer DF, Gannon J, Deely E. Reliability of Crack Detection for Pre-Construction Condition Assessments. J Infrastructure Systems 2010 ASCE; 16(2): 129-37.
[8] Wehr A, Lohr U. Airborne Laser Scanning-An Introduction and Overview. ISPRS Journal of Photogrammetry and Remote Sensing 1999; 54(2-3): 68-82.
[9] Ackermann F. Airborne Laser Scanning-Present Status and Future Expectations. ISPRS Journal of Photogrammetry and Remote Sensing 1999; 54(2-3), 64-7
[10] Lemmens M. (2009). "Terrestrial Laser Scanners." August 2009
[11] Huising EJ, Gomes Pereira LM. Errors and Accuracy Estimates of Laser Data Acquired by Various Laser Scanning Systems for Topographic Applications. ISPRS Journal of Photogrammetry and Remote Sensing 1998; 53(5): 245-261
[12] Hinks T, Carr H, Laefer DF. Flight Optimization Algorithms for Aerial LiDAR Capture for Urban Infrastructure Model Generation. J Comp Civ Eng 2009, ASCE, Special Issue, Graphical 3D Visualization in Arch., Eng., & Construction; 23 (6): 330-9.
[13] Baltsavias EP. Airborne Laser Scanning: Basic Relations and Formulas. ISPRS J Photogrammetry & Remote Sensing 1999; 54: 199-214.
[14] Wenisch P, van Treeck C, Borrmann A, Rank E, Wenisch O. Computational Steering on Distributed Systems: Indoor Comfort Simulations as a Case Study of Interactive CFD on Supercomputers. Int J Parallel, Emergent & Distributed Systems 2007; 22 (4): 275-91.
[15] Laefer, DF, Hinks, T, Carr, H. New Possibilities for Damage Prediction Form Tunnel Subsidence Using Aerial LiDAR data. *ISSMGE: Geotechnical Challenges in Megacities,* Moscow, June 7-10, 2010, 2: 622-9.
[16] Laefer DF, Carr H, Morrish S, Kalkan E. Opportunities and Impediments to the use of 3D Laser Scanning for Adjacent Excavations. *GeoCongress 2006: Geotechnical Engineering in the Information Technology Age,* Atlanta, GA, Feb 26-March 1, 2006.
[17] Hoffman CM, Rossignac JR. A Road Map to Solid Modeling. IEEE Trans on Visualization and Comput graphics 1996; 2(1): 3-10.
[18] Requicha AA, Rossignac JR. Solid Modeling and Beyond. IEEE Computer Graphics and Applications 1992; 12 (5): 31-44.
[19] Goldman R. An Integrated Introduction to Computer Graphics and Geometric Modeling, CRC Press, Taylor and Francis, New York 2009.
[20] Velayutham P. An Efficient Algorithm for Converting Polyhedral Objects with Winged-Edge Data Structure to Octree Data Structure. MSc Thesis, Ind. Eng., University of Cincinnati, USA, 2005.
[21] ANSYS Academic Research V12. Help System, Theory Reference for ANSYS and ANSYS Workbench.
[22] Bruenet P, Navaro I. Solid Representation and Operation Using Extended Octrees. ACM Transactions on Graphics (TOG) 1990; 9(2): 170-97.
[23] Jackins CL, Tanimoto SL. Oct-trees and Their Use in Representing Three-Dimensional Objects. Comput Graphics Image Proc 1980; 14(3): 249-70.
[24] Meagher D. Geometric Modeling Using Octree Encoding. Comput Graphics Image Proc 1982; 19(2): 129-47.
[25] Samet H, and Webber RE. Hierarchical Data Structures and Algorithms for Computer Graphics. Part I. *IEEE Comput. Graph. Appl. 1988;* 8(3), 48-68.
[26] Weyrich T, Pauly M, Heinzle S, Keiser R, Scandella S, Gross M. Post-processing of Scanned 3D Surface Data. *Proc Eurographics Symposium on Point-Based Graphics,* Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, June 2-4, 2004; 85-94.
[27] Gopi M, Krishnan S. A Fast and Efficient Projection-Based Approach for Surface Reconstruction. High Performance Comput Graphics, Multimed Visualisation 2002; 1 (1): 1-12.
[28] Krishnan R, Das A, Gurumoorthy B. Octree encoding of B-rep based objects. Computers & Graphics 1996; 20(1), 107-114.
[29] Mencl R, Muller H. Interpolation and Approximation of Surfaces from Three-Dimensional Scattered Data Point. Proceeding Dagstuhl '97, Scientific Visualization 1999; 223-232.
[30] Lorensen WE, Cline HE. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *SIGGRAPH '87 Proceedings*; 21(4): 163-9.
[30] Sharman J. The Marching Cubes Algorithm.
Available at: http://www.exaflop.org/docs/marchcubes/index.html (Accessed on: March 3, 2010).
[31] Olson ES. System and Method for Surface Reconstruction from an Unstructured Point Set. US2009/0171627A1 (**2009**)
[32] Azernikov S, Fischer A. Efficient Surface Reconstruction Method for Distributed CAD. Comp-

Aided Design 2004; 36: 799-808.
[33] Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W. Surface reconstruction from unorganized points. *ACM SIGGRAPH 1992*, 71-8.
[34] Curless B, Levoy M. A Volumetric Method for Building Complex Models From Range Images. *SIGGRAPH '96 Proceedings*: 303-12.
[35] Bernardini, F., Rushmeier, H. The 3D Model Acquisition Pipeline. Computer Graphics forum 2002; 21(2): 149-72.
[36] Gross H, Thoennessen U. Extraction of Lines from Laser Point Clouds. *International Archives of Photogrammetry, Remote Senning and Spatial Information Sciences*, Bonn, Germany, 86-91.
[37] Schuster HF. Segmentation of Lidar Data Using the Tensor Voting Framework. *International Archives of Photogrammetry Remote Sensing And Spatial Information Sciences 2004;* 35(3), 1073-8.
[38] Pulli K, Duchamp T, Hoppe H, McDonald J, Shapiro L, Stuetzle W. Robust Meshes from Multiple Range Maps. International Conference on Recent Advances in 3-D Digital Imaging and Modeling 1997; 205-211.
[39] Dalmasso P, Nerino R. Hierarchical 3D Surface Reconstruction Based on Radial Basis Functions. *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, 574-579.
[40] Osher S, Thiyanaranam P. Method and Apparatus for Accurate Compression and Decompression of Three-Dimensional Point Cloud Data. US 2010/0239178 A1 (**2010**)
[41] Edelsbrunner H, Mücke EP. Three-dimensional alpha shapes. ACM Transactions on Graphics (TOG) 1994; 13 (1), 43-72.
[41] Pu S, Vosselman G. Extracting Windows from Terrestrial Laser Scanning. ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007, Espoo, Finland, 320-5.
[42] Boulaassal H, Landes T, and Grussenmeyer P. Automatic Extraction of Planar Clusters and their Contours on Building Façades Recorded by Terrestrial Laser Scanner. International Journal of Architectural Computing 2009; 7(1), 1-20.
[43] Boissonnat JD. Geometric Structures for Three Dimensional Shape Representation. ACM Transactions on Graphics 1984; 3(4): 266-286.
[44] Pu S, Vosselman G. Automatic Extraction of Building Features from Terrestrial Laser Scanning. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 2006; Dresden, Germany, CD-ROOM.
[45] Secord J and Zakhor A. Tree Detection in Urban Regions Using Aerial LiDAR and Image Data. Journal of Geoscience and Remote Sensing Letters, IEEE 2007; 4(2), 196-200
[46] Matei BCM, Samarasekera S, Kim JY, Karney CFF, Sawhney HS, Kumar R. Building Segmentation for Densely Built Urban Regions Using Aerial Lidar Data. US 20090310867/A1 (**2009**)
[47] Bernardini F, Mittleman J, Rushmeier H. The Ball-Pivoting Algorithm for Surface Reconstruction. IEEE Transactions on Visualization and Computer Graphics 1999; 5(4), 349-59.
[48] Hook TV. Real-time shaded NC milling display. ACM Signnraph 1986; 15-20.
[49] Várady T, Facellob M, Teréka Z. Automatic Extraction of Surface Structures in Digital Shape Reconstruction. Comp Aided Design 2007; 39(5): 379-88.
[50] Hinks T, Carr H, Laefer DF, Morvan Y, O'Sullivan C, Truong-Hong L. Robust Building Outline Extraction. PTO 56793223 (**2009**).
[51] Zhou K, Huang X, Guo B. User-guided Surface Reconstruction. US 2010/0085353A1 (**2010**)
[52] Bradeley C. Currie B. Advances in the Field of Reverse Engineering. Computer-Aided Design and Application 2005; 2(5), 697-706
[53]Kubit. Point clouds in AutoCAD. Available at: http://www.kubitusa.com/CAD/Products/PointCloud_Laser_Scanner_AutoCAD/laser_scanner_AutoCAD.php (Accessed on: October 8, 2009).
[54] Rapidform. Rapidform XOR/ Redesign. Available at: http://www.rapidform.com/portal/default/Products/index?Category=Products_RapidformXORRedesign_XORSub1. (Accessed on: November 26, 2009).
[55] Autodesk Catalogue. Available at: http://autodeskcatalog.com/2010/03/29/for-over-ten-years-kubit-has-provided-autocad-based-solutions-for-capturing-as-built-conditions-and-making-surveying-simple-for-any-cad-user/ (Accessed on: March 29, 2010).
[56] Schueremans L, Genechten BV. The Use of 3D Laser Scanning in Assessing Safety of Masonry Vaults-A Case Study on The Church of Saint-Jacobs. Optics and Lasers in Eng 2009; 47: 329-35.
[57] Smars P, Schueremans L, Van Balen K. Monitoring the Dismantlement of Four Flying Buttresses. *Proc SAHC 2006*; 3:1421-8.
[58] Young PG, Beresford-West TBH, Coward SRL, Notarberardino B, Walker B, Abdul-Aziz A. An Efficient Approach to Converting Three-Dimensional Image Data Into Highly Accurate Computational

Models. Philosophical Trans Royal Society 2008; 366: 3155-73.
[59] Viceconti M, Zannoni C, Pierotti L. TRI2SOLID: An Application of Reverse Engineering Methods to The Creation of CAD Models of Bone Segments. Comp Methods & Programs in Biomedicine 1998; 56(3): 211-20.
[60] Schmitt J, Meiforth J, Lengsfeld M. Development of A Hybrid Finite Element Model for Individual Simulation of Intertrochanteric Osteotomies. Medical Eng & Physics 2001; 23(8): 529-39.
[61] Wirtz DC, Pandorf T, Portheine F, Randermacher K, Schiffers N, Prescher A, et al. Concept and Development of An Orthotropic FE Model of The Proximal Femur. J Biomech 2003; 36(2): 289-93.
[62] Lorenz G. System and Method for Automatic Mesh Generation From A System-Level MEMS Design. US7131105 (**2006**).
[63] ClearEdge3D. ClearEdge3D Website. Available at: http://www.clearedge3d.com/edgewise/ [Accessed on: December 29, 2009].
[64] Arefi H, Engels J, Hahn M, Mayer H. Levels of Detail in 3D Building Reconstruction from LiDAR Data. *The Int Arch Photogrammetry, Remote Sensing & Spatial Info Sci,* Beijing, China, July 3-11, 2008, 37-B3b: 485-490.
[65] Geomagic, 2009. Geomagic Website. Available at:
http://www.geomagic.com/en/products/studio/features.php [Accessed on November 26th, 2010].
[66] Trimble G200. Trimble Website. Available at: http://www.trimble.com [Accessed on Nov 25th, 2010].
[67] RealWorks Survey. Trimble Website. Available at: http://www.trimble.com/realworks.shtml [Accessed on Nov 25th, 2010].
[68] Xu B, Yu W. Body Surface Imaging. US 2010/0277571 A1 (**2010**)
[69] Lubowiecka I, Armestob J, Ariasb P, Lorenzob H. FEM Modeling of Structures Based on Close Range Digital Photogrammetry. Automation in Construction 2009; 18: 559-69.
[70] Ripperda N, Brenner C. Application of a Formal Grammar to Facade Reconstruction in Semiautomatic and Automatic Environments. *12th AGILE International Conference on Geographic Information Science*. Hannover, Germany, June 2-5th, 2009.
[71] Werner T, Zisserman A. New Techniques for Automated Architectural Reconstruction from Photographs. Proceedings of the 7th European Conference on Computer Vision-Part II, Springer-Verlag. Copenhagen, Denmark, 27 May - 2 June, 2002.
[72] Muller P, Zeng G, Wonka P, Gool L V. Image-Based Procedural Modeling of Facades." *ACM Trans. Graph* 2007; 26(3), 85.
[73] Quadling M, Quadling H, Li Y, Tchouprakov A. 3D Photogrammetry Using Projected Patterns. US20080101688 (**2008**) and WO2008052092 (**2007**).
[74] Becker S, Haala N. Grammar Supported Facade Reconstruction From Mobile Lidar Mapping." *CMRT09, IAPRS*. Paris, France, Sep 3-4th, 229-234.
[75] Pu S, Vosselman G. Knowledge Based Reconstruction of Building Models from Terrestrial Laser Scanning Data. *ISPRS Journal of Photogrammetry and Remote Sensing 2009;* 64(6), 575-584.