ETH zürich Research Collection

Conference Paper

From point cloud to surface the modeling and visualization problem

Author(s):
Remondino, Fabio

Publication Date:
2003

Permanent Link:

https://doi.org/10.3929/ethz-a-004655782 →

Rights / License:

In Copyright - Non-Commercial Use Permitted →

This page was generated automatically upon download from the ETH Zurich Research Collection. For more information please consult the Terms of use.

ETH Library

International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XXXIV-5/W10. International Workshop on Visualization and Animation of Reality-based 3D Models, 24-28 February 2003, Tarasp-Vulpera, Switzerland

FROM POINT CLOUD TO SURFACE: THE MODELING AND VISUALIZATION PROBLEM

Remondino Fabio
Institute of Geodesy and Photogrammetry
Swiss Federal Institute of Technology
ETH Zurich, Switzerland
fabio@geod.baug.ethz.ch

Commission V – WG 6

ABSTRACT
In this paper we address all the problems and solutions of converting a measured point cloud into a realistic 3D polygonal model that can satisfy high modeling and visualization demands. Close range photogrammetry deals since many years with manual or automatic image measurements. Now 3D scanners are also becoming a standard source for input data in many application areas, providing for millions of points. As a consequence, the problem of generating high quality polygonal meshes of objects from unorganized point clouds is receiving more and more attention. After reviewing the different 3D shape techniques for surface reconstruction, we will report the full process of converting a usually unstructured point cloud into a consistent polygonal model. Some triangulation algorithms, modeling methods and visualization techniques are also described and different examples are presented.

Keywords: Modeling, Reconstruction, Visualization

1. INTRODUCTION

The reconstruction of precise surfaces from unorganized point clouds derived from laser scanner data or photogrammetric image measurements is a very hard problem, not completely solved and problematic in case of incomplete, noisy and sparse data. The generation of polygonal models that can satisfy high modeling and visualization demands is required in different applications, like video-games, movies, virtual reality applications, e-commerce and other graphics applications. The goal is always to find a way to create a computer model of an object which best fit the reality. Polygons are usually the ideal way to accurately represent the results of measurements, providing an optimal surface description. While the generation of digital terrain models has a long tradition and has found efficient solutions, the correct modeling of closed surfaces or free-form objects is of recent nature, a not completely solved problem and still an important issue investigated in many research activities.
The wide range of applications from which the data may emerge (e.g. manufacturing, reverse engineering, cultural heritage, architecture) implies that the data can have quite different properties that must be considered in the solution of the surface interpolation problem. Many methods have been developed [Mencl, 2001] to create a regular and continuous (triangular) mesh representation from a point cloud. Then given the polygonal surface, various techniques can be used for post-processing operations (smoothing, texturing) and for the visualization of the 3D model [Patias, 2001a].
This paper comes from our experience and research in the photogrammetric field. This work wants to put together almost all the current methods and techniques for modeling and visualization of 3D scenes, in particular obtained from measured point clouds. After a short overview of common modeling and visualization terms, a review of the different techniques to recover 3D shapes is reported (section 3). In section 4 the process of converting a usually unstructured point cloud into a consistent polygonal model ("triangulation") is described. Finally visualization packages and techniques are reviewed and discussed.

2. TERMINOLOGY

Nowadays it is very common to read and hear words like Rendering, Shading or NURBS and maybe we are not really familiar with the correct meaning. Before going into details regarding the modeling and visualization aspects, a short list of important terms, which does not cover all the aspects of the presented subjected, is reported. More information are provided in [Patias, 2001a].

- Aliasing: it is the undersampling of a signal (e.g. geometry or texture) that causes artifacts; in case of raster devices, like a computer screen, lines appears jagged when a vector image is drawn. The minimization of the appearance of jagged edges of polygons or lines during the visualization is called anti-aliasing.
- Breakline: feature line or polyline representing a ridge or some other feature (e.g. folds) that the user wishes to preserve in a mesh made up of polygonal elements. Therefore a breakline is a singularity on a surface, like an edge to which the polygons should conform to, i.e., not intersect.
- Level of detail (LoD): it is the amount of information (detail) or complexity of any object that is displayed at any particular time. The LoD is usually a function of the distance of the object from the viewer.
- Mesh: it is a collection of triangular (or quadrilateral) contiguous, non-overlapping faces joined together along their edges. A mesh therefore contains vertices, edges and faces and its easiest representation is a single face. Sometimes it is also called TIN, Triangulated Irregular Network. Finite element methods are generally used to generate a surface mesh.
- Modeling: the (mathematical) construction and computer implementation of an object, by defining points in a 3 dimensional array. This array is based on the X, Y and Z axis of geometric space. Then, different sets of these points are mathematically 'joined' by e.g. lines, to create polygons and the polygons joined to create objects. The simplest result is usually displayed as a wireframe model.

- Open GL: a 3D graphics programmer interface, initially design by SGI and now developed by several companies, to improve the performances of graphical hardware supporting the Open GL standard. Direct3D (by Microsoft) is another standard implementation.
- Rendering: a usually realistic drawing of 3D objects using computer technologies. In order to render an object, certain properties like transparency, colour, diffuse or specular reflection and refraction must be assigned to it and to the surrounding environment. Two common rendering techniques are called *ray tracing* (it renders the model object by object (or better pixel by pixel), testing if a ray intersects any object in the scene and calculating the light intensity in that particular direction) and *scanline* rendering (it renders the image of an object as a series of horizontal or vertical lines). Ray tracing is not suitable for real-time visualization. Scanline does not produce as realistic results as ray tracing, but it is frequently used in animation packages where the image quality of the single frames is not so important. Rendering can be geometry-based or image-based (usually called 'Texture mapping').
- Shading: it is the assignment of surface properties to an object. They are colour, normal information, reflectance, transparency and lighting model.
- Splines: A piecewise polynomial function that can have a locally very simple form but at the same time be globally flexible and smooth. Splines are very useful for modeling arbitrary functions and are used extensively in computer graphics for free-form curves and surfaces representation. A class of parametric curves and surfaces is the Non-Uniform Rational B-Spline (NURBS) curve or surface. They are the generalization of non-rational B-splines, which are basis of polynomial function based on rational Bézier curves.
- Surface: a compact, connected, orientable 2 or 3D manifold, possibly with boundary. A surface without boundary will be called a closed surface. A surface can be geometrically represented in implicit form (locus of the solution of the function F(x,y,z)=0) or parametric form (a collection of parametric patches properly joined together). Surfaces, which cannot be described in an analytic form, are called free-form surfaces.

3. 3D SHAPE TECHNIQUES FOR 3D MODEL RECONSTRUCTION

The methods to recover 3D shapes and models can be divided into two classes:

1. systems based on objects measurements;
2. systems that do not use measurements.

3.1 Systems based on measurements

These techniques (based on images or on active 3D sensors) can be mainly divided in 2 categories:

- methods based on triangulation (Figure 1): they use image measurements [Remondino, 2003; D'Apuzzo 2002], structured light [Maas, 1992; Gartner et al, 1995; Sablatnig et al., 1997], coded light [Wahl, 1984], laser light [11, Sequeira et al., 1999].
- methods that do not require correspondences: they estimate surface normals instead of 3D data. Examples are shape from shading [Horn et al., 1989], shape from texture [Kender, 1978], shape from specularity [Healey et al., 1987], shape from contour (medical applications) [Meyers et al., 1992; Minoru et al., 1993; Ulupinar et al. 1995], shape from 2D edge gradients [Winkelbach et al., 2001]. Other approaches use active 3D sensors with the time-of-flight principle (in particular for large structures) [48].

Passive image-based methods (e.g. photogrammetry or computer vision) acquire 3D measurements from single images (section 6) or multi-stations; they use projective geometry [Pollefeys, 2000] or perspective camera model [Gruen et al., 2002c; Remondino, 2003]; they are very portable and the sensors are not expensive. On the other hand, 3D active sensors (mainly laser scanners) are becoming a common approach for objects or scenes recording and a standard source for geometric input data. They provide for millions of points and often also for the associated colour. But these sensors are quite expensive, designed for specific applications and depend on the reflective characteristics of the surface. A review of optical methods and active range sensors for 3D measurements is presented in [Beraldin et al., 2000; Chen et al, 2000; Blais, 2003].

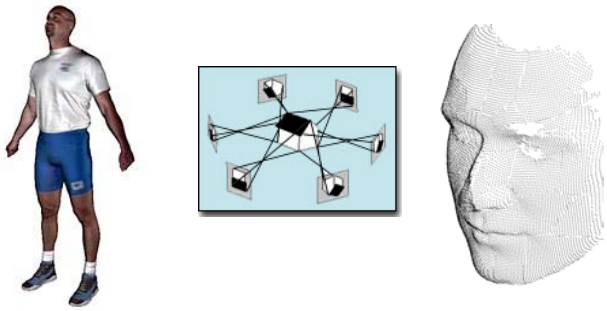

Figure 1: 3D model of human produced with a Cyberware Body Scanner [11] (left). A bundle adjustment solution for multi-image photogrammetric reconstruction (center; Source: Photomodeler [31]). A point cloud generated with a matching algorithm on 5 images (right) [D'Apuzzo, 2002].

3.2 Systems not based on measurements

They are commercial computer animation software (Table 2) that allow the generation of 3D models starting from simple elements like polygonal boxes. They generally subdivide and smooth polygons using 3D splines; they do not use any measurement providing for realistic results (Figure 2). They are mainly used in the animation community for movies and video-games [44].

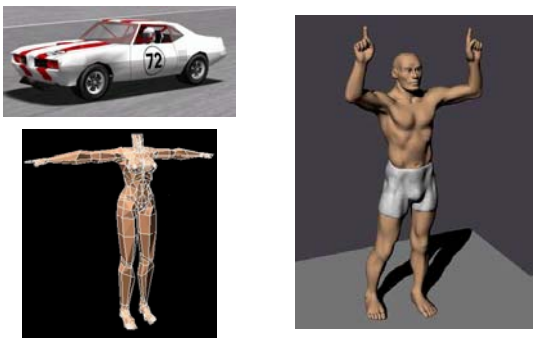

Figure 2: 3D model created with computer animation software. Left: 3D Studio Max [3], right: Lightwave [23].

3.3 Photogrammetric reconstruction and modeling process

Photogrammetry, by definition, obtains reliable measurements and 3D models by means of photographs. It deals since many years with the 3D reconstruction of objects from one or more images: even if it mostly requires precise calibration and orientation procedures, reliable packages are now available (e.g. Photomodeler [31], ShapeCapture [38], Australis [7]). The overall process, described in Figure 3, consists of few well-known steps:

- Design (sensor and network geometry)
- Measurements (point clouds, lines, etc.)
- Structuring/Modeling (geometry, texture)
- Visualization/Analysis of the results

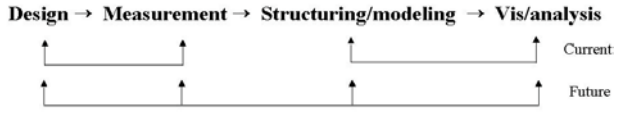

Figure 3: Photogrammetric reconstruction process as presented in [Gruen, 2002b]

Nowadays, the recovery of the sensor (and network) geometry and the measurement phase are mainly separated from the modeling and visualization part. But in many applications [Gruen, et al., 2002c] this gap must be bridged in order to perform correct measurements and recover realistic 3D models.

The measurement step can be performed with manual or automatic procedures. Automated photogrammetric matching algorithms can produce very dense point clouds, but mismatches, irrelevant points and missing parts could be present in the results, requiring a post-processing check of the data. These automated procedures usually do not take into consideration the geometrical conditions of the surface's object and mainly work with smoothing constraints: therefore is often quite difficult to turn randomly generated point clouds into polygonal structures of high quality and without losing important information. Nowadays 3D laser scanners are also becoming a common source of input data, providing for big data sets. Therefore the modeling problem of these unorganized point clouds is receiving great attention.

On the other hand, if the measurements are done in manual or semi-automatic mode, there is a higher reliability of the measures but a smaller number of points that describe the object; moreover it is very important for the operator to understand the functional behaviour of the following 3D modeling software to perform correct measurements. In this context an on-line modeler that project onto the stereomodel the generated mesh to control the agreement between measurements and the structure of the object would be very useful.

After the measurements, modeling and visualization of the results can be performed with different techniques, as described in section 4 and section 5.

4. SURFACE RECONSTRUCTION AND MODELING

In this section the generation of surfaces (in particular free-form and closed surfaces) from point clouds produced with laser scanner or stereo measurements is described. Other methods, e.g. based on shape from texture, shape from shadow or surface from contours are not considered here.

The goal of surface reconstruction can be stated as follows: given a set of sample points P assumed to lie on or near an unknown surface S, create a surface model S' approximating S. A surface reconstruction procedure cannot guarantee the recovering of S exactly, since we have information about S only through a finite set of sample points. Sometime additional information of the surface (e.g. breaklines) can be available and, in general, as the sampling density increases, the output result S' is more likely topologically correct and converges to the original surface S. A good sample should be dense in detailed area and sparse in featureless parts. Usually if the input data does not satisfy certain properties required by the algorithm (like good distribution and density, little noise), the reconstruction program produces incorrect or maybe impossible results. Therefore, the correct reconstruction method depends on the application and for each application, the right algorithm (program) must be used.

Why is the surface reconstruction a difficult problem? Firstly the measured points are usually unorganized and often noisy; moreover the surface can be arbitrary, with unknown topological type and with sharp features. Therefore the reconstruction method must infer the correct geometry, topology and features just from a finite set of sample points.

4.1 Classification of the reconstruction algorithms

It is very complicated to classify all the reconstruction methods. The universe of algorithms is quite large but in this section we attempt to report them according to some categories, like 'used approach', 'type of data' or 'representation'. Some algorithms could belong to different groups, therefore we list them only once. In [Boissonat et al., 2002] the approaches for surface meshing of unorganized point clouds are organized in four categories, while an overview of the algorithms, updated to 1998, is also presented in [Mencl et al., 1998]

Our first and very general classification is done according to the *quality (type) of the input data*:

- Unorganized point clouds: algorithms working on unorganized data have no other information on the input data except their spatial position. They do not use any assumption on the object geometry and therefore, before generating a polygonal surface, they usually structure the points according to their coherence. They need a good distribution of the input data and if the points are not uniformly distributed they easily fail.
- Structured point clouds: algorithms based on structured data can take into account additional information of the points (e.g. breaklines).

A further distinction can be done according to their *spatial subdivision*:

- Surface oriented algorithms do not distinguish between open and closed surfaces. Most of the available algorithms belong to this group [Hoppe et al., 1992, Mencl, 2001].
- Volume oriented approaches work in particular with closed surfaces and generally are based on the Delaunay tetrahedrization of the given set of sample points [Boissonat, 1984; Isselhard et al., 1997; Curless et al., 1996].

Another classification is based on the *type of representation* of the surface:

- Parametric representation: these methods represent the surface as a number of parametric surface patches, described by parametric equations. Multiple patches may then be

pieced together to form a continuous surface. Examples of parametric representations include B-spline, Bezier curves, and Coons patches [Terzopulos, 1988].
- Implicit representation: these methods try to find a smooth function that passes through all positions where the implicit function evaluates to some specified value (usually zero) [Gotsman et al., 1998].
- Simplicial representation: in this representation the surface is a collection of simple entities including points, edges and triangles. This group includes Alpha shapes [Edelsbrunner et al., 1994] and the Crusts algorithm [Amenta et al., 1998].
Always according to the way of representation, approximated or interpolated surfaces can be generated:
- Approximated surfaces do not always contain all the original points, but points as near as possible to them. They can use a distance function (shortest distance of a point in space from the generated surface) to estimate the correct mesh [Hoppe et al., 1992]. In this group we can also insert the warping–based surface reconstruction (they deform an initial surface so that it gives a good approximation of the given set of points) [Muraki, 1991] and the implicit surface fitting algorithms (they fit e.g. piecewise polynomial functions to the given set of points) [Moore et al., 1990].
- Interpolated surfaces are instead used when precise models are requested: all the input data are used and a correct connection of them is necessary [Dey et al., 2001].

Finally, we can classify the reconstruction methods according to the different *assumptions of the algorithm*:
- Algorithms assuming fixed topological type: they usually assume that the topological type of the surface is known a priori (e.g. plane, cylinder or sphere) [Brinkley, 1985; Hastie et al., 1989]
- Algorithms exploiting structure or orientation information: many surface reconstruction algorithms exploit the structure of the data for the surface reconstruction. For example, in case of multiple scans, they can use the adjacency relationship of the data within each range image [Merrian, 1992]. Other reconstruction methods instead use knowledge of the orientation of the surface that is supplied with the data. For example, if the points are obtained from volumetric data, the gradient of these data can provide orientation information useful for the reconstruction [Miller et al., 1991].

4.2 From points to surface

The conversion of the measured data into a consistent polygonal surface is generally based on four steps:
1. pre-processing: in this phase erroneous data are eliminated or points are sampled to reduce the computation time (section 4.3);
2. determination of the global topology of the object's surface: the neighbourhood relations between adjacent parts of the surface has to be derived. This operation typically needs some global sorting step and the consideration of possible 'constraints' (e.g. breaklines), mainly to preserve special features (like edges);
3. generation of the polygonal surface: triangular (or tetrahedral) meshes are created satisfying certain quality requirements, e.g. limit on the meshes element size, no intersection of breaklines, etc. (section 4.4);
4. post-processing: when the model is created, editing operations are commonly applied to refine and perfect the polygonal surface (section 4.5).

4.3 Pre-processing operations

Editing operations on the measured points are very important before generating a triangular surface. The pre-processing operations usually are:
- data sampling, based on the curvature of the points or uniformly apply. In case of scanner data, this step is mandatory in order to reduce the input redundancy (down-sampling) and to remove a certain amount of errors introduced because of to the scanning device limitations [Schreiber, 1993; Floater et al., 1998].
- noise reduction and outliers rejection: statistical methods are applied taking into consideration the surface curvature and trying to preserve the measured features. In case of image matching results, wrong correspondences can be removed automatically [Fua et al., 1992; Borghese et al., 2000] or manually with visual inspection.
- holes filling: gaps in the point clouds are closed adding (manually or automatically) new points and using the curvature and density of the surrounding points.

4.4 Triangulation or mesh generation

It is the core part of almost all reconstruction programs. See the book by [Edelsbrunner, 2001] for a good and recent introduction to the topic. A triangulation converts the given set of points into a consistent polygonal model (mesh). This operation partitions the input data into simplices and usually generates vertices, edges and faces (representing the analysed surface) that meet only at shared edges. Finite element methods are used to discretize the measured domain by dividing it into many small ‘elements’, typically triangles or quadrilaterals in two dimensions and tetrahedra in three dimensions. An optimal triangulation is defined measuring angles, edge lengths, height or area of the elements while the error of the finite element approximations is usually related to the minimum angle of the elements. The vertices of the triangulation can be exactly the input points or extra points, called Steiner points, which are inserted to create a more optimal mesh [Bern et al., 1992]. Triangulation can be performed in 2D or in 3D, according to the geometry of the input data.

4.4.1 2D Triangulation

The input domain is a polygonal region of the plane and, as result, triangles that intersect only at shared edges and vertices are generated. A well known construction method is the Delaunay triangulation (DT) that simultaneously optimize several of the previous mentioned quality measures. [Fortune, 1992].

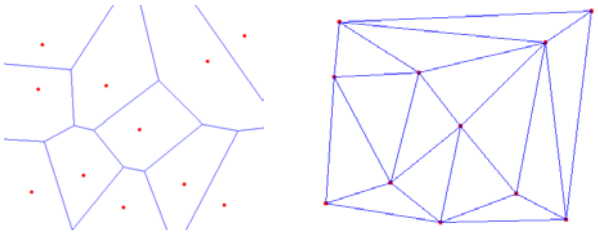

Figure 4: Voronoi diagram (left) and Delaunay triangulation (right) of the same set of points. In Voronoi, each region consists of the part of the plane nearest to that node. Connecting the nodes of the Voronoi cells that have common boundaries forms the Delaunay triangles.

Delaunay criterion ensures that no vertex lies within the interior of any of the circumcircles of the triangles in the network. DT of a given set of point is the dual of the Voronoi diagram (also called the Thiessen or Dirichlet tessellation), as shown in Figure 4.

4.4.1.1 2.5D Triangulation

The input data is a set of points P in a plane along with a real and unique elevation function f(x,y) at each point (x,y) ∈ P. A 2.5D triangulation creates a linear function F interpolating P and defined on the region bounded by the convex hull of P. For each point p in P, F(p) is the weighted average of the elevation of the vertices of the triangle that contains p. Usually Delaunay triangulation is used as interpolation function. According to the data structure, regularly or almost randomly distributed, the generated surface is called elevation grid or TIN (Triangulated Irregular Network) model.

4.4.1.2 Surfaces for 3D models

The input data is always a set of point P in R^3, but no more restricted on a plane; therefore the elevation function f(x,y) is no more unique. The input set is also called unorganized point cloud.

4.4.2 3D Triangulation

The triangulation in 3D is called tetrahedralization or tetrahedrization. A tetrahedralization is a partition of the input domain into a collection of tetrahedra that meet only at shared faces (vertices, edges or triangles). Tetrahedralization results are much more complicated than a 2D triangulation. The types of input domains could be simple polyhedron (sphere), non-simple polyhedron (torus) or point clouds.

4.5 Post-processing operations

The created polygons usually need some refinements to correct imperfections or errors in the surface. These operations (mainly manually) vary from single triangles editing to surface corrections:

- edges correction: faces can be splitted (divided in two parts), moved to another location or contracted.
- triangles insertion: holes can be filled constructing polygonal structures that respect the surrounding area; incomplete meshes can also be repaired with radial basis function [Carr et al., 2001] or with volumetric approach [Curless et al., 1996].
- polygons editing: the number of polygons can be reduced, preserving the shape of the object or fixing the boundary points (other automatic operations, used in particular for compression of polygonal structures, are described in section 5.4). The polygonal model can also be improved adding new vertices and adjusting the coordinates of existing vertices. Moreover spikes can be removed with smooth functions.

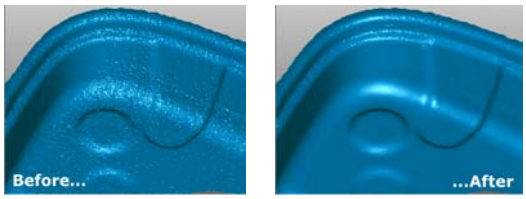

Figure 5: Smoothing of spikes, noise or bad polygons on a polygonal model [14].

4.6 Modeling software

Modeling software are packages that perform all the operations described in the precedent sections. Polygons are usually the ideal way to accurately represent the results of measurements, providing an optimal surface description. Therefore, with the improving of 3D measurement techniques (in particular 3D laser scanners), tools producing polygonal surfaces from point clouds (we called them 'reverse engineering software') are becoming more and more necessary for accurate representations of organized or unorganized 3D data (Table 1).

Paraforms	3D Reshaper	Geomagic	Cyclone
FarField	Imageware Surfacer	Polyworks	Solid Works
Rapidform	Spatial Analyzer	AutoCAD	MicroStation

Table 1: Commercial 'reverse engineering' and CAD software for modeling applications.

But at the same time, powerful 3D modeling and rendering packages (we call them 'computer animation software'), mainly spline-based, including tools for 3D object modeling (from pre-defined primitives like cubes, sphere, cones, etc.), lighting controls and texture mapping are increasing their popularity, expecially in the graphic community (Table 2). They generally do not allow importing 3D point clouds and their final goal is usually the animation of the created model.

Softimage 3D	Poser	Extreme 3D	3D Shockwave
Easymodel	Amira	Cinema 4D	Animation Master
Rhinoceros	AC3D	I-Sculpt	Corel Dream 3D
3D Studio Max	Maya	Lightwave	Model Magic 3D
Vue	Bryce	RenderMan	World Builder

Table 2: Some computer animation packages.

There are also some packages and libraries, mostly developed in the universities, that are free (or shareware) on the Internet for public download and test, like Cocone (it allows importing point clouds) [9], Amapi [5], Blender [8], GLUT or GL Space [17], Imagine [20], VolPack (it imports data sampled on a 3D regular grid) [45]. They usually produce realistic results but are not suitable for very big data sets.

All the computer animation software (Table 2) provides animation functions while, generally, 'reverse engineering' software cannot produce videos.

More information concerning modeling software, discussion forums, books, 3D laser scanners and tutorials is available at [40, 44].

5. VISUALIZATION

In many applications like particle tracking, fog, clouds or water visualization and with large amount of points, the data can be visualized by just drawing all the samples [Reeves, 1983; Pfister et al., 2000]. However, for some objects (and not very dense point clouds) this technique does not give good results and does not provide realistic visualization. Moreover the visualization of a 3D model is often the only product of interest for the external world and remains the only possible contact with the model. Therefore a realistic and accurate visualization is often required.

In the photogrammetric community, the first attempts in the visualization of 3D models were done at the beginning of the '90. Small objects (e.g. architectural models, cars, human

faces) were displayed in wireframe format [Gruen et al., 1988], or using CAD packages [Streilein et al., 1991], while terrain models were visualized in perspective wireframe models with draping of orthophotos or orthophotomaps [Baltsavias et.al., 1991]. Nowadays, with the increasing of the computer memories, shade and texture are added to all the models, but in order to accurate visualize big data sets, much information contained in photogrammetric models is often reduced [Patias, 2001b]. The consequences are that the accuracy of the data is lost (many tools use single precision files) as well as the geo-referencing (most of the software have their own coordinate systems) and that high resolution textures are unusable (because of the control on the Level of Detail). On the other end, low accuracy in the visualization does not attract the end-users and cannot justify the high cost of producing the photogrammetric 3D model.

5.1 Ways to display 3D models

After the creation of a triangular mesh (section 4), the results are usually visualized, according to the used package and the requirements, in the following manners:

- Wireframe mode: it is the easiest way of representing a 3D object. It consists of points, lines and curves and describes only the edges in a transparent drawing, without texture or shading information. This technique is mainly used in computer-aided design (CAD) packages.
- Shaded mode: it is based on the optical theory (Lambert's Cosine Law) which states that the brightness of any small area (polygon) of a perfectly diffuse undulating surface arises as the cosine of the angle of incident parallel light. There are many different shading algorithms, the most well know are *flat shading* and *smooth shading*. The key difference between flat and smooth shading is in the way that the normals are used. Flat shading simply assigns each triangle a normal vector and the lighting is done individually on each face. For smooth shading, the surface normals of the surrounding faces are averaged and at each vertex is assigned a normal. Flat (or constant) shading is valid for small object and if the source light and the viewer are at infinity; for high detail levels, a great number of flat shaded polygons is required, therefore it is of little value for realism. Smooth (or interpolated) shading can be applied with many algorithms, but the two "classic" approaches are Gouraud and Phong. Gouraud shading specifies a colour for each vertex and polygon and then intermediate colours are generated along each edge by interpolation between the vertices. Phong shading requires a normal interpolation for each pixel and so is quite prohibitive and time consuming for real-time use.
- Textured mode: it is used for photorealistic visualization of the 3D models (image-based rendering). Texture mapping in its simplest form involves a single texture (image, orthophoto) being mapped onto the surface composed of one or more polygons. When mapping an image onto an object, the colour of the object at each pixel is modified by the corresponding colour derived from the texture. Compared to flat shading, texture mapping can reduce the number of used polygons, but increase the 'weight' of the model (in terms of visualization). A lot of memory is required for rendering realistic 3D texturized model: therefore a new graphic interface, called AGP (Accelerated Graphics Port), has been developed [18]. AGP allows the texture to be stored in a main memory more powerful than video memory and can speed up the transfer of large textures between memory, CPU and video adapter.

In case of 3D terrain model (DTM) other common methods of representation are the contour maps, the colour shaded models (hypsometric shading) or the slope maps. In a contour map, contour lines are generated and displayed from the terrain models by intersecting horizontal planes with the network. In a colour shaded model, the height information of the model are displayed with colours.
In general, creating realistic 3D models (shaded or texturized) helps to visualize the final result much better than a wireframe representation. With a wireframe, because no hidden surface removal is performed, it is not always clear to distinguish e.g. from which viewpoint we are looking at the model. Instead shading and rendering can greatly enhance the realism of the model. To decide which type of model to produce, we have to consider different factors, like time, hardware and needs. If the time is limited or the hardware cannot stand big files, detailed shaded models might not be necessary. On the other hand, presentations or virtual flights do require texturized models.

Figure 6: A closed view of a 3D model of the Bamiyan Buddha represented in wireframe, flat shaded and textured mode [Gruen et al., 2002c].

5.2 Visualization packages and 3D file formats

Visualization tools and software are available in different forms, freeware or commercial, for big models or small data sets (Table 3): some packages are also able to insert information like text documents, hyperlinks, sounds and movies. Some of them are programmed just to display 3D scenes and with few “edit” functions (VRML viewers, Skyline [39], etc.). On the other hand computer animation software (Table 2) always allow visualization and animation functions but cannot import many 3D formats. Until few years ago CAD packages could mainly generate 3D volume models and visualize them in wireframe or texturized mode; but nowadays they can also render real world free-form objects as accurately as graphical packages do, like 3D Studio Max [3] or Maya [25] (Figure 7).

AutoCAD	ArcInfo	Virtual GIS
Skyline	Cyberwalk	Macromedia Director
TerraVista	ArcView	Idrisi
Vituozo	TerrainView	SolidWorks
Vrwave	*Scene Viewer*	*Cosmo Player*
Inventor	*Vis5D+*	*GeomView*

Table 3: Some visualization packages. The italic software is free on the Internet. Other commercial solutions available for terrain model visualization are provided by [43].

According to the type of model (terrain or small object) and the required visualization (animation, flight through, static display) different packages can be used: some are specific

and usable only with particular models, others can read different 3D formats (Table 4). For a first use, it is very difficult to select the correct package, as it needs different tests and practical work on it with different data sets to choose the right product.
The visualization software can be classified using different quality criteria or parameters (interactivity, antialiasing, Level of Detail); a common one divides the packages, given a particular hardware configuration, on the basis of their real-time performance:
- high-end visualization tools: they are commercial packages that allow all the visualization and animation needs also on small laptops (e.g. Skyline [39]);
- middle class tools: they still can render big models, but with slow real-time performances (e.g. Virtual GIS [12]);
- low-end tools: mainly freeware on the Internet, they can display in real-time only small data sets (e.g. Cosmo Player [10]).

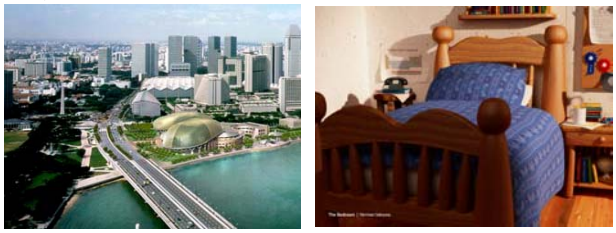

Figure 7: 3D models rendered with a CAD package and a computer animation software. Left: MicroStation result rendered with ray tracing [26]. Right: 3D Studio Max result of a bedroom [H.Saksono, 3].

A big difficulty at the moment is the translation and interchange of 3D data between modeling/visualization packages. Each (modeling) software has always its own (binary) format. Even if they allow exporting files in other formats, the generated 3D file often cannot be imported in other packages. Usually the only workable export format from a CAD package is the DXF, which has no texture, requires large storage and often produces considerable errors. Some converters of 3D graphics file formats are available (PolyTrans [33], 3DWin [4], or [24]), but a standardization of the many 3D formats should be done to facilitate the exchanges of the models. Little standardizations have been done, like the VRML format [46] (AVI or MPEG for video), allowing easily exchanges of 3D models e.g. on the Internet.
A good and complete visualization package should provide the possibility to import, display and export different 3D formats (Table 4) as well as image texture data. This flexibility is usually provided by some university self-developed packages.

3DMax	VRML	Lightwave	Solid Works
OpenFlight	DXF	Inventor	Stereolithography STL
Apple 3DMF	IGES	DirectX	Adobe Illustrator
Cinema 4D	Maya	Shockwave	Wavefront OBJ/MTL

Table 4: Most common 3D file formats.

5.3 Animation and Interactive Navigation of 3D Models

An *animation* of a 3D model can be defined as a sequence of 'snapped' frames, obtained from a visualization package and joined together. We can animate either the camera or the 3D model (all model or its single parts independently) or both. The model is viewed from different viewpoints and each view contributes to the sequence. The first animation were created using CAD software (AutoCAD, MicroStation) and wireframe models. Then vector data were also inserted to texture the objects and only in the middle of the 90's, with the growth of computer memories, real images were used to produce photo-realistic 3D animated models.
Nowadays, according to standard definition and format, different viewers and software are available to display animated 3D models. AVI and MPEG formats are becoming very popular as well as free packages and viewers for these formats, giving a big push to animations and virtual flight creators. Another common standard for interactive and not pre-constructed visualization of 3D scenes is the *3D navigation* (interactive or walk-trough). The user-interactivity is achieved with many tools (languages, libraries and software), as shortly presented afterwards. New tools become available every day, therefore a complete survey is not realistic and would not be complete: a good overview is presented in [Patias, 2001b].
A problem of most visualization packages is the aliasing effect: it appears during the rendering and the animation of a scene, because the spatial frequency of the scene is higher then Nyquist frequency of the monitor. Terrain visualization packages do not always provide antialiasing function for geometry and textures while computer animation packages allow antialiasing control.

5.3.1 VRML (Virtual Reality Modeling Language) [46]

It is one of the most common 3D interactive navigation language and also file format. The current standard is VRML 97, usually known as VRML 2. It is an ISO standard format for representing 3D models (the representation of geographic data in VRML is called GeoVRML). VRML language allows to create different 3D scenes through which the user can navigate using a (web) browser plug-in, like Cosmo Player (for web browsers), Vrweb or Vrwave (for Sun/SGI/Linux), TerraForm or SolidView or WorldView (for Windows PC). VRML gives also the possibility to store some key-positions defined in different viewpoints: then automatic flights through the 3D model can be generated according to the defined viewpoints.

5.3.2 Open Graphics Library (OpenGL) [29]

OpenGL is mainly a software interface to graphic hardware. The libraries consist of several procedures and functions to easily draw objects in 2D and 3D and control their rendering via hardware accelerators. Open GL is developed by Silicon Graphics (Cosmo Open GL) and Microsoft (Microsoft Open GL). It provides interactive animation and antialiasing function, but no portable 3D models. It can be combined with programming languages like C, Fortran, Delphi.

5.3.3 Skyline [39]

It is a full package, which does not require user programming and permit all visualization and animation needs. It is a high-end terrain visualization tool, allowing large data-set management, texturing with high-resolution images and real-time fly-through of geographic data.

5.4 Compression of geometric data sets

New measurement technologies usually produce large amounts of data which are then converted into meshes with hundred of millions of polygons. For some data sets of these sizes, common algorithms for real-time visualization,

animation and transmission of these meshes are not practical. Therefore, in the last decade, a trend, in particular in the graphic community, has been the study of sampled representation of these big data sets. The requirements were:

- speed up the transmission of big geometric models;
- improve the rendering and visualization performances;
- reduce the cost of storage and memory without losing important information.

Usually two types of information are encoded in the created meshes: the geometrical information (i.e. the position of the vertices in the space and the surface normals) and the topological information (i.e. the mesh connectivity and the relations between the faces). Based on these two information and on the needs listed before, many compression algorithms have been proposed, mainly for triangular meshes; these methods are based on:

- compression of the geometry of the data: they try to improve the storage of the numerical information of the meshes (positions of the vertices, normals, colours, etc.) or they look for an efficient encoding of the mesh topology [Deering, 1995; Taubin et al., 1998; De Floriani et al., 1998].
- control of the Level-of-Detail (LoD): it is done in particular for visualization purposes. The LoD varies smoothly throughout the scene and the rendering depends on the current position of the model. A control on the LoD allows view-dependent refinement of the meshes so that details that are not visible (occluded primitives or back faces) are not shown [Hoppe, 1998; Kumar et al., 1996; Duchaineau et al., 1997]. The control can also be performed on the texture using image pyramids (or MIP-mapping [27]), impostors [Karner et al., 2001; Wimmer et al., 2001] or a view-dependent texture mapping [Visnovcova et al., 2001].
- mesh optimization, filtering and decimation: they simplify the meshes removing vertices, edges and triangles. They can iteratively remove vertices that do not pass certain distance/angle criterion or collapse edges into a unique vertex [Hoppe, 1997]. Other approaches are based on vertex clustering [Rossignac et al., 1993], wiener filtering [Alexa, 2002] and wavelets [Gross et al., 1996].
- point rendering: it is applied in particular for point clouds visualization and it works by displaying a smaller number of primitives.

Between the proposed solutions, QSplat [Rusin et al., 2000] showed good results in terms of performances, real-time display, compression and quality of the rendered models. It was built at the Stanford University, during the large project "The Digital Michelangelo" and with the goal of constructing a software that could be useable also on low-end computers without special 3D support. It is free on the Internet and demonstrated real-time handling of 100 million to 1 billion points. QSplat is a point-based rendering system able to render the same object using shapes of different complexity for the splat element and different transparency to avoid aliasing (jagged edges) effects [36]. Other public domain tools for mesh simplification (also called 'Multiresolution Modeling Software') are VTK (Visualization Toolkit [47]), Jade [22], PolyReduce [32], Qslim [35]; commercial tools are Cosmo Worlds and OpenGL Optimizer for SGI [30], IBM Interaction Accelerator [19], Innovmetrix IMCompress [21]. For general overviews and further links, visit [42].

6. THE RECONSTRUCTION AND MODELING PROBLEM FROM ONE IMAGE

The generation of 3D geometry of a scene can be also performed from a single perspective image. Different cues like texture, focus and shading can be identified and used to process the image and infer 3D shapes. A classical approach projects a stripe pattern on the object and measures edge direction or surface normal. These systems require a camera and a projector and are mainly used for small and near objects (e.g. statues, faces) [Vuylsteke et al. 1990]. In case of man-made objects, planes, right angles and parallel lines must be contained in the image: the vanishing points and lines are determined and afterwards 3D (affine or metric) reconstruction can be performed using geometric image invariants [Van der Heuvel, 1998; Criminisi et al., 1999]. In case of free-form curved surfaces (e.g. persons or landscapes), given a set of user-specified constraints on the local shape of the scene, a smooth 3D surface can be generated. The user specifies point positions, normals, silhouette and RoI and a constrained optimisation algorithm compute the best surface that satisfies these constraints [Zhang et al., 2001].

Figure 8: Two example of 3D reconstruction from a single image. Upper: 3D model of a building reconstructed using vanishing points and straight lines information [Criminisi et al., 1999]. Lower: Landscape 3D model reconstructed by [Zhang et al., 2001] using specific constraints on the shape of the scene.

7. CONCLUSIONS

In this paper a description of all the processes to convert a measured point cloud into a polygonal surface and then visualize it with a computer has been presented. An almost complete overview of the modeling/visualization solutions has been reported, including also spline-based packages, but many other solutions are probably implemented in various labs and we are not informed of them. Even if the concepts and algorithms of the modeling software are quite straightforward, the performance of the software is strictly related to the implementation and to the hardware. Moreover all the existing software for modeling and visualize 3D objects are specific for certain data sets. Commercial 'reverse engineering' packages do not produce correct meshes without dense point clouds. On the other hand visualization tools can do nothing to improve a badly modeled scene and the

rendering can get worse the results if antialiasing or level of detail control are not available.

ACKNOWLEDGEMENT

The author would like to thank Andreas Roditakis of the Institute of Geodesy and Photogrammetry, ETH Zurich, for the useful talks and suggestions on the modeling topic.

REFERENCES

Adamson, A., Alexa, M., 2003: Ray Tracing Point Set Surfaces. Int. Conference on Shape Modeling and Applications. In press

Amenta, N., et al., 1998: New Voronoi Based Surface Reconstruction Algorithm. ACM Proc. of Siggraph, pp. 415-422

Alexa, M., 2002: Wiener Filtering of Meshes. Proc. of Shape Modeling International, pp. 51-57

Baltsavias, E.P., Gruen, A., Meister, M., 1991: A Digital Orthophoto Workstation. ACSM/ASPRS/AUTO-CARTO 10th Annual Meeting

Beraldin, J.A, Blais, F, et al., 2000: Active 3D sensing. Scuola Normale Superiore Pisa, Centro di Ricerche Informatiche per I Beni Culurali, Quaderni 10, 21 pp

Bern, M., Eppstein D., 1992: Mesh Generation and Optimal Triangulation. Computing in Euclidean Geometry, D.Z. Du, F.Hwang, eds, World Scientific, Lecture Notes Series on Computing, Vol. 1, pp. 23-90

Blais, F., 2003: A review of 20 years of range sensor development. Videometrics VII, SPIE Proc., Vol. 5013, pp. 62-76

Boissonnat, J.D., 1984: Geometric structures for three-dimensional shape representation. ACM Transactions on Graphics, 3(4), pp.266–286

Borghese A., Ferrari S., 2000: A Portable Modular System for Automatic Acquisition of 3-D Objects. IEEE Trans. on Instrumentation and Measurement, Vol. 49, No.5, pp. 1128-1136

Brinkley, J.F., 1985: Knowledge-driven ultrasonic three-dimensional organ modeling. IEEE Transaction on PAMI, Vol.7, No.4, pp.431–441

Carr, J.C., et al., 2001: Reconstruction and Representation of 3D Obje cts with Radial Basis Functions. ACM Proc. of Siggraph, pp.67-76

Chen, F., et al., 2000: Overview of three-dimensional shape measurement using optical methods. Optic. Eng., Vol. 39, pp. 10-22

Criminisi, A, et al., 1999: Single View Metrology. Proc. International Conference on Computer Vision, pp.434-442

Curless, B., Levoy, M., 1996: A volumetric method for building complex models from range images. ACM Proc. of Siggraph, pp.303-312.

D'Apuzzo, N., 2002. Modeling human faces with multi-image photogrammetry. Three-Dimensional Image Capture and Applications V, Proc. of SPIE, Vol. 4661, San Jose, USA, pp. 191-197

Deering, M, 1995: Geometry Compression. ACM Proc. Siggraph, pp. 13-20

De Floriani, L., et al., 1998: Efficient implementation of multi-triangulations. Proceedings IEEE Visualization

Dey T.K., Giesen, J., 2001: Detecting undersampling in surface reconstruction. Proc. 17th Symposium of Computational Geometry, pp.257-263

Duchaineau, M, et al., 1997: ROAMing Terrain: Real-Time Optimally Adapting Meshes. Proceedings IEEE Visualization

Edelsbrunner H., Mucke, E., 1994: Three Dimensional Alpha Shapes. ACM Transactions on Graphics, Vol. 13, No 1

Edelsbrunner, H., 2001: Geometry and Topology for Mesh Generation, volume 6 of Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, UK.

Floater, M., Iske, A., 1998: Thinning Algorithms for Scattered Data Interpolation. BIT Numerical Mathematics, Vol. 38, No 4, pp. 705-720

Fortune, S., 1992: Voronoi diagrams and Delaunay triangulation. Computing in Euclidean Geometry, D.Z. Du, F.Hwang, eds, World Scientific, Lecture Notes Series on Computing, Vol. 1

Fua P., Sander, P.T., 1992: Reconstructing surfaces from unstructured 3D points. Proceedings Image Understanding Workshop, pp 615–625, San Diego, CA, USA

Gartner, H., Lehle, P., Tiziani, H.J., 1995: New, high efficient, binary codes for structured light methods. SPIE Proceedings, Vol. 2599, pp. 4-13

Gross, M., et al., 1996: Efficient Triangular Surface Approximations using Wavelets and Quadtree Structures. IEEE Transaction on Visual and Computer Graphics, Vol.2, No. 2, pp. 130-144.

Gotsman, C., Keren, D., 1998: Tight Fitting of Convex Polyhedral Shapes. Int. Journal of Shape Modeling, 4(3-4), pp.111-126

Gruen, A., Baltsavias, E.P., 1888: Automatic 3-D measurement of human faces with CCD-cameras. SPIE Proceedings "Biostereometrics ´88

Gruen, A., Wang, X., 2002a: Integration of landscape and city modeling: The pre-hispanic site Xochicalco. Int. Workshop on Visualization and Animation of Landscape. Kunming, Thailand. (34-5/W3), ISSN 1682-1777

Gruen, A., 2002b: Return of the Buddha - New Paradigms in Photogrammetric Modeling. Keynote Address, ISPRS Commission V Symposium, Corfu. Available at: http://www.photogrammetry.ethz.ch/general/persons/AG_pub/keynotes_corfu/index.htm

Gruen, A., Remondino, F., Zhang, L., 2002c: Reconstruction of the Great Buddha of Bamiyan, Afghanistan. Int. Arch. of Photogrammetry and Remote Sensing, Vol. 34, part 5, pp. 363-368

Hastie, T., Stuetzle W., 1989: Principal curves. JASA, Vol. 84, pp. 502–516

Healey, G., Binford, T.O., 1987: Local Shape from Specularity. Proc. ICCV, London

Hoppe, H., et al., 1992: Surface reconstruction from unorganized points. ACM Proc. of Siggraph, pp.71-78

Hoppe, H., 1997: View-Dependent Refinement of Progressive Meshes. ACM Proc. Siggraph, pp.189-198

Hoppe, H., 1998: Smooth View-Dependent Level of Detail Control and its Application to Terrain Rendering. IEEE Proc. Visualization, pp.25-42

Horn, B.K.P., Brooks, M.J., 1989: Shape from Shading. MIT Cambridge

Karner, K., et al., 2001: Virtual Habitat: Models of the Urban Outdoors. In Gruen, Baltsavias, Van Gool (editors): Workshop on 'Automated Extraction of Man-Made Objects from Aerial and Space Images' (III), Balkema Publishers, pp 393-402, Ascona, Switzerland

Kender, J.R., 1978: Shape from Texture. Proc. DARPA IU Workshop

Kumar, S., Et al., 1996: Hierarchical Back-Face Computation. Proc. Eurographics Rendering Workshop, pp.231-240

Isselhard, F., et al, 1997: Polyhedral reconstruction of 3D objects by tetrahedra removal. Technical report No. 288/97, Fachbereich Informatik, University of Kaiserslautern, Germany

Maas, H.G., 1992: Robust Automatic Surface Reconstruction with Structured Light. Int. Archives of Photogrammetry and Remote Sensing, Vol. 24, Part B5, pp. 102-107

Mencl, R., Mueller, H., 1998: Interpolation and Approximations of surfaces from Three-Dimensional Scattered Data Points. State of the Art Report for Eurographics Conference, Lisbon, Portugal

Mencl, R., 2001: Reconstruction of Surfaces from Unorganized 3D Points Clouds. PhD Thesis, Dortmund University, Germany

Merriam, M., 1992: Experience with the cyberware 3D digitizer. Proceedings NCGA, pp. 125–133

Meyers, D., et al., 1992: Surfaces from contours. ACM Transactions on Graphics, Vol. 11, No. 3, pp.228–258

Miller, J.V., et al., 1991: Geometrically deformed models: A method for extracting closed geometric models from volume data. ACM Proc. of Siggraph, Vol. 25, No. 4, pp. 217–226

Minoru, A., Nakamura T., 1993: Cylindrical Shape from Contour and Shading without Knowledge of Lighting Conditions or Surface Albedo. IPSJ Journal, Vol. 34, No. 5

Moore D., Warren, J., 1990: Approximation of dense scattered data using algebraic surfaces. TR 90-135, Rice University, USA

Muraki, S, 1991: Volumetric shape description of range data using "blobby model". ACM Proc. of Siggraph, pp. 217–226

Patias, P., 2001a: Photogrammetry and Visualization. Technical Report, ETH Zurich. Available at: http://www.photogrammetry.ethz.ch/research/guest.html

Patias, P., 2001b: Visualization Issues in Photogrammetry. Proc. of the 3rd International Seminar on New Development in Digital Photogrammetry, Gifu, Japan, pp. 4-6

Pfister, H., et al., 2000: Surfels: Surface Elements as Rendering Primitives. ACM Proc. Siggraph, pp. 335-342

Pollefeys, M., 2000: Tutorial on 3-D modeling from images. Tutorial at ECCV 2000. Available at: www.esat.kuleuven.ac.be/~pollefey/publications/tutorial.pdf

Reeves, W.T., 1983: Particle Systems: A Technique for Modeling a Class of Fuzzy Objects. Computer Graphics, Vol. 17, No 3, pp. 359-376

Remondino, 2003: 3D Reconstruction of Static Human Body with a Digital Camera. Videometrics VII, SPIE Proc., Vol. 5013, pp. 38-45

Rossignac, J., Borrel, P., 1993: Multi-resolution 3D approximation for rendering complex scenes. In Geometric Modeling in Computer Graphics. Falcidieno, Kunii, eds. Springer Verlag, pp. 455–465

Rusinkiewicz, S, Levoy, M., 2000: Qsplat: a Multiresolution Point Rendering System for Large Meshes. ACM Proc. of Siggraph, pp.343-352

Sablatnig, R., Menard, C., 1997: 3D Reconstruction of Archaeological Pottery using Profile Primitives. Sarris N., Strintzis M.G., eds, Proc. of International Workshop on Synthetic-Natural Hybrid Coding and Three-Dimensional Imaging, pp. 93-96

Schreiber, T., 1993: Clustering for Data Reduction and Approximation. Int. Conf. on Computer Graphics and Visualization, S. Petersburg, Russia

Sequeira V., Ng K., et al. 1999: Automated reconstruction of 3D models from real environments. ISPRS Journal for Photogrammetry and Remote Sensing, 54(1), pp. 1-22

Streilein, A., Beyer, H., 1991: Development of a digital system for architectural photogrammetry. Proceedings of XIV International Symposium of CIPA

Taubin, G., Rossignac, J., 1998: Geometric Compression through Topological Surgery. ACM Trans. on Graphics, Vol.17, No 2

Terzopoulos, D., 1988: The Computation of Visible Surface Representation. IEEE Transactions on PAMI, Vol. 10, No 4

Ulupinar F., Nevatia R., 1995: Shape from Contour: Straight Homogeneous Generalized Cylinders and Constant Cross Section Generalized Cylinders. Trans. PAMI Vol. 17, No. 2

Van den Heuvel, F.A., 1998. 3D reconstruction from a single image using geometric constraints. ISPRS Journal for Photogrammetry and Remote Sensing, 53(6), pp. 354-368

Visnovcova, J., Zhang, L., Gruen, A., 2001: Generating a 3D model of a Bayon tower using non-metric imagery. Int. Archives of Photogrammetry and Remote Sensing, Vol. XXXIV, Part 5/W1, pp. 30-39.

Vuylsteke, P., Oosterlinck, A., 1990: Range Image Acquisition with a Single Binary-Encoded Light Pattern. IEEE Transaction on PAMI, Vol. 12, No. 2

Wahl, F.M., 1984: A Coded Light Approach for 3-Dimensional Vision. IBM Research Report, RZ 1452

Wimmer, M., et al., 2001: Point-based Impostors for Real-Time Visualization. Proceedings of Eurographics Workshop on Rendering. Berlin: Springer

Winkelbach, S., Wahl, F.M., 2001: Shape from 2D Edge Gradient. Pattern Recognition, Lecture Notes in Computer Science 2191, Springer

Zhang, L., et al., 2001: Single View Modeling of Free-Form Scenes. CVPR Proceedings. Available at: http://grail.cs.washington.edu/projects/svm/

URL References (January 2003):

1. 3D Modeling & Rendering, Online Course Resource: http://online.caup.washington.edu/courses/arch411/00HomePage/0.default.html
2. 3D Reshaper: http://www.technodigit.com/english/
3. 3D Studio Max: http://www.3dmax.com or http://www.discreet.com/products/3dsmax/
4. 3DWin: http://www.tb-software.com/
5. Amapi: http://www.eovia.com/
6. Animation Master: http://www.glasspalace.fi/am/
7. Australis: http://www.sli.unimelb.edu.au/australis/
8. Blender: http://www.blender3d.org/
9. Cocone: www.cis.ohio-state.edu/~tamaldey/cocone.html
10. Cosmo Player: http://www.cai.com/cosmo/
11. Cyberware: http://www.cyberware.com/
12. ERDAS: http://www.erdas.com/software/main.asp
13. FarField: http://www.farfieldtechnology.com/
14. Geomagic Studio: http://www.geomagic.com/
15. Graphic and Visualization Software: http://www.cs.purdue.edu/homes/sun/Resources/software_graphics.html
16. GeoVRML: http://www.geovrml.org/
17. GL Space: sal.kachinatech.com/E/3/GL-SPACE.html
18. Hardware Guide: http://www6.tomshardware.com/
19. IBM: http://www.research.ibm.com/3dix
20. Imagine: http://www.coolfun.com
21. Innovmetric: http://www.innovmetric.com/home.html
22. Jade: www.vislab.usyd.edu.au/users/manuals/3dgraphics/jade/index.html
23. Lightwave: http://www.lightwave3d.com/
24. List of translator: http://www.web3d.org/vrml/vft.htm
25. Maya: www.aliaswavefront.com/en/products/maya/index.shtml
26. Microstation: http://www.bentley.com/
27. MIP Mapping: http://www.visionengineer.com/comp/mip_mapping.shtml
28. Modeling, Rendering and Animation Software: http://www.huntfor.com/3d/software/modeling.htm
29. Open GL: http://www.opengl.org
30. OpenGL Optimizer: http://www.sgi.com/software/optimizer/
31. Photomodeler: http://www.photomodeler.com
32. PolyReduce: http://hendrix.imm.dtu.dk/software/
33. PolyTrans: http://www.okino.com/
34. Polyworks: http://www.innovmetric.com/
35. Qslim: http://www-2.cs.cmu.edu/afs/cs/user/garland/www/scape/index.html
36. QSplat: http://graphics.stanford.edu/software/qsplat/
37. RenderMan: https://renderman.pixar.com/
38. ShapeCapture: http://www.shapecapture.com
39. Skyline: http://www.skylinesoft.com/
40. Simple3D: http://www.simple3d.com.
41. Spatial Analyzer: http://www.kinematics.com/sa_intro.htm
42. Surface Approximation and Multiresolution Modeling: http://www.ams.sunysb.edu/~jsbm/surfapprox.html or http://almond.srv.cs.cmu.edu/afs/cs/user/garland/www/multires/software.html
43. Terrain Visualization Software list: http://www.tec.army.mil/TD/tvd/survey/survey_toc.html
44. Ultimate 3D Links: http://www.3DLinks.com/
45. VolPack: http://graphics.stanford.edu/software/volpack/
46. VRML: http://www.web3d.org/vrml/vrml.htm
47. VTK: http://public.kitware.com/VTK/
48. Zoller-Foehlich GmbH: http://www.zofre.de