1

Basic Background in 3D Object Processing

Guillaume Lavoué

1.1 3D Representation and Models

This chapter details the different 3D representation models, commonly used in computer graphics and 3D modeling. We distinguish three main representation schemes: polygonal meshes; surface-based models, such as parametric, implicit and subdivision surfaces; and volumetric models including primitive-based models (superquadric, hyperquadric), voxel representation and constructive solid geometry. For each of these 3D representation schemes, we analyze the advantages and drawbacks in terms of modeling, regarding different applications.

The point-based representation, which has been recently introduced in computer graphics, is not within the scope of this book. Recent reviews can be found in Kobbelt and Botsch (2004) and Alexa *et al.* (2004).

1.1.1 Basic Notions of 3D Object Representation

Now 3D objects are more complex to handle than other multimedia data, such as audio signals or 2D images, since there exist many different representations for such objects.

For instance, a 2D image has a unique and rather simple representation: a 2D grid ($n \times n$) composed of n^2 elements (named *pixels*) each containing a color value or a gray level. The different devices and techniques which produce digital images (digital cameras, scanners, etc.) all provide the same representation.

For 3D models there are different kinds of representation: an object can be represented on a 3D grid like a digital image, or in 3D Euclidean space. In the latter case, the object can be expressed by a single equation (like algebraic implicit surfaces), by a set of facets representing its boundary surface or by a set of mathematical surfaces.

The main difficulties with a 3D object are that:

- The different sources of 3D data (tomography, laser scanning) do not produce the same representations.
- The different applications (computer-aided design, medical) do not consider the same representations.
- Changing from one representation to another is quite complex and often constitutes open problems.

Figure 1.1 illustrates several representations of the 3D object *Bunny*.

From a real-world object, a laser scanner produces a set of points in 3D space, each defined by its coordinates x, y and z. Figure 1.1(a) illustrates a cloud of points representing the object *Bunny*. This point-based representation, provided by scanners, is not efficient for computing physical or geometrical properties or displaying on a screen. Indeed there are no neighborhood relations between 3D points, neither is there any surface or volumetric information. Therefore these representations are often converted to polygonal meshes and particularly *triangular* meshes (see Figure 1.1(b)). With this model, the object is represented by its boundary surface which is composed of a set of planar faces (often triangles). More precisely, a polygonal mesh contains a set of 3D points (the *vertices*) which are linked by *edges* to form a set of polygonal *facets*. Many *surface reconstruction* techniques exist to produce a triangular mesh starting from a cloud of points issuing from a 3D scanner.

Polygonal meshes can represent open or closed surfaces from arbitrary topology, with a precision that depends on the number of vertices and facets. Intersection, collision detection or rendering algorithms are simple and fast with this model, since manipulating planar faces (and particularly triangles) is also simple (linear algebra). This rapidity is particularly useful

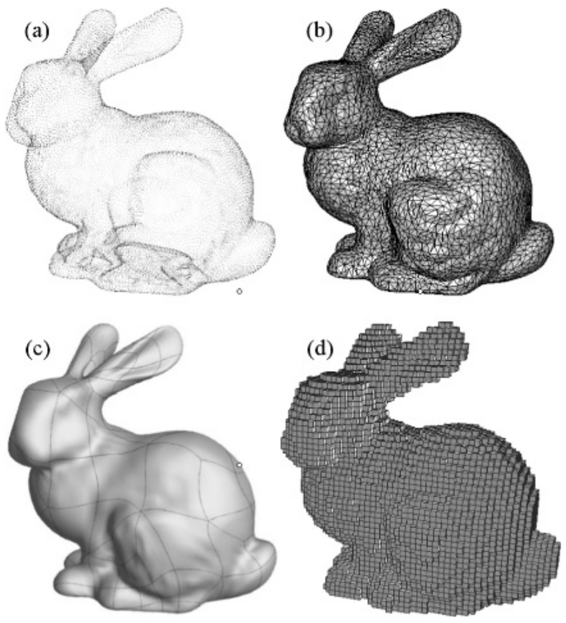


Figure 1.1 Different representations of *Bunny*: (a) a cloud of points, (b) a triangular mesh, (c) a set of parametric surfaces, (d) a set of voxels

for videogames. These benefits make this model the most widespread representation for 3D objects.

However, polygonal meshes have some limitations. This model is intrinsically *discrete* since the number of vertices and facets depends on the expected precision, thus a high precision can lead to a huge amount of data. Moreover, the definition of the shape is very local and thus applying a global deformation or manually creating a shape facet by facet is quite difficult.

The need for a 3D model adapted for modeling and conception has led to the appearance of parametric surfaces. This family of 3D surfaces (including Bézier, B-spline and NURBS surfaces) is particularly used for computer-aided design (CAD), manufacturing (CAM) and engineering (CAE). This model enables mathematically exact surfaces to be defined contrary to polygonal meshes which only represent an approximation. Since these surfaces are defined on a parametric domain, they cannot represent a shape with arbitrary topology, hence a 3D object is often represented by a patchwork of parametric surfaces. For instance, *Bunny* from Figure 1.1 is modeled with 153 bicubic B-spline patches.

These surfaces have very strong tangential and curvature continuity properties that make them quite useful for design. Moreover, they are mathematically complete and allow modeling of a large variety of shapes. Since they are defined only by a few sets of control points, instead of a dense set of vertices, they are also much more compact than polygonal meshes. The main drawback is that they are more complex to manipulate than a set of triangles.

Polygonal meshes and parametric surfaces are boundary representations (BREP) since they model an object only by its boundary. Several domains (especially medical imaging) need the interior data of a 3D object and thus consider a volumetric model and particularly the *discrete* representation. This model does not represent the 3D object in Euclidean space, but in a 3D grid similar to the 2D image representation. Each element of the grid is a voxel (short for volumetric pixel). Thus the object is represented by the set of voxels constituting its volume. A voxel can contain a boolean value (*in* the object or *outside* the object) or other information like local densities. Medical devices (RMI for instance) often produce such volumetric data to describe the interior of an organ. Figure 1.1(d) illustrates the *voxelized Bunny*, in a $50 \times 50 \times 50$ grid.

1.1.2 Polygonal Meshes

A 3D polygonal mesh is defined by a set of plane polygons (see Figure 1.2). Thus, such a model contains three kinds of *elements*: *vertices*, *edges* and *faces* (the most used are triangles). Additional attributes can be attached to the vertices, such as normal vectors, color or texture information. A polygonal mesh consists of two kinds of information: the *geometry* and the *connectivity*. The geometry describes the position of the vertices in 3D space and the connectivity describes how to connect these positions, i.e. the relationship between the mesh elements. These relations specify, for each face, the edges and vertices which compose it, and for each vertex, the incident edges and faces. The *valence* of a vertex is the number of its incident edges and the *degree* of a face is its number of edges (see Figure 1.2). A polygonal mesh is called *manifold* if each of its edges belongs to one or two faces (one in the case of a border). Manifold meshes present strong geometrical properties, and thus are considered in almost all the existing mesh compression methods. A manifold mesh is associated with its Euler–Poincaré characteristic χ:

$$\chi = v - e + f \tag{1.1}$$

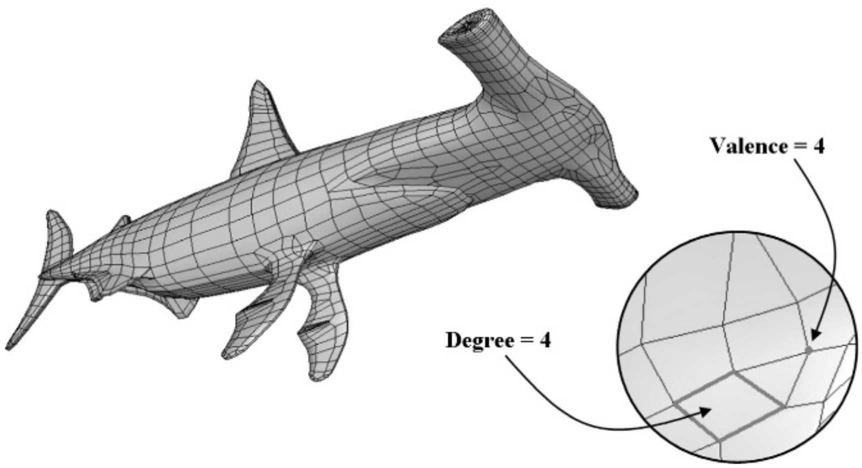


Figure 1.2 Polygonal mesh example (2560 vertices, 2562 faces, 5120 edges), illustrating the *valence* of a vertex and the *degree* of a face

with v, e and f, respectively, the numbers of vertices, edges and faces of the mesh. The Euler–Poincaré characteristic is related to the genus of the corresponding surface, according to the following equation:

$$g = \frac{2c - b - \chi}{2} \tag{1.2}$$

where c is the number of connected components and b the number of borders. The genus of a surface describes its topological complexity; it corresponds to the maximum number of closed curves, without common points, which can be traced inside this surface without disconnecting it (i.e. the complement to these curves remains connected). Basically the genus is the number of *handles* of the mesh; a sphere and a torus are respectively of genuses 0 and 1. Figure 1.3 illustrates a genus-65 model.

The polygonal mesh model is by far the most popular in computer graphics and 3D modeling. Its algebraic simplicity (linear algebra) largely facilitates algorithms of intersection, collision detection or rendering. Moreover, polygonal meshes have a high capacity of description; indeed any complex object of arbitrary topology can be modeled by a polygonal mesh, provided there is enough memory space available. Finally, many techniques generate a polygonal mesh starting from other geometrical models (parametric surfaces, implicit surfaces, discrete models, etc.). This model representation largely

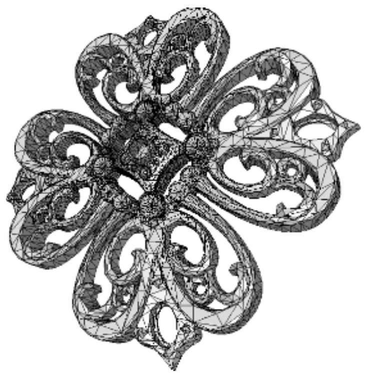


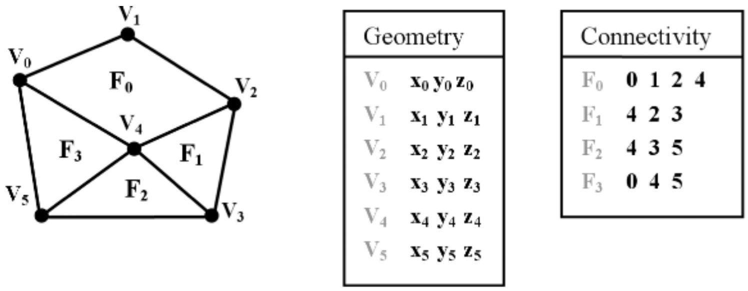
Figure 1.3 Example of a genus-65 3D mesh (8830 vertices, 17919 faces). Courtesy of SensAble technologies by the AIM@SHAPE Shape Repository

dominates the various 3D interchange formats, such as VRML and MPEG-4 (2002).

Nevertheless this model has several limitations: it is a *discrete* representation, because it depends on the targeted scale. Indeed, a polygonal mesh only possesses a C^0 continuity, and thus cannot represent exactly a smooth surface, only an approximation whose precision (linked to the number of polygons) depends on the targeted scale and application (rendering, collision detection, etc.). Consequently, this model can become quite heavy in terms of the amount of data if the expected precision is high, or if the object to be represented is complex.

The standard mesh format (used in the VRML standard, for instance) is as follows. The geometry is represented by a list of coordinates indexed over the vertices and the connectivity is described by the list of faces, each of them being represented by a cyclic list of indices of its incident vertices. Figure 1.4 illustrates this standard representation. The file contains the coordinates of the vertices V_0, V_1, V_2, V_3, V_4, V_5 (the *geometry*) and for each face F_0, F_1, F_2, F_3, the indices of the corresponding vertices (the *connectivity*).

Although this coding method offers the advantage of simplicity, encoded data are very redundant since vertices and edges are referred to several times. In the binary format, for a triangular mesh, each of the three coordinates of each vertex is basically encoded by a *float* (4 bytes) and for each face, each of the three vertex indices is encoded by an *integer* (4 bytes). If n_s is the number of vertices and n_f the number of faces, then the size of the file



Geometry	
V_0	$x_0\, y_0\, z_0$
V_1	$x_1\, y_1\, z_1$
V_2	$x_2\, y_2\, z_2$
V_3	$x_3\, y_3\, z_3$
V_4	$x_4\, y_4\, z_4$
V_5	$x_5\, y_5\, z_5$

Connectivity	
F_0	0 1 2 4
F_1	4 2 3
F_2	4 3 5
F_3	0 4 5

Figure 1.4 Example of standard representation for a simple mesh, containing six vertices and four faces

is $(n_f + n_s) \times 4 \times 3$ bytes. For a regular mesh, for which $n_f \approx 2 \times n_s$, we have to encode $3 \times 4 \times 3 = 36$ bytes (or 288 bits) per vertex.

1.1.3 Surface-based Models

In this class of representation, a 3D object is represented by its boundary, defined by one or more pieces of a surface. We detail here the three main representation schemes comprising this family: parametric surfaces, implicit surfaces and subdivision surfaces. Polygonal meshes also belong to this class of representation, but due to its predominance in computer graphics, we have devoted a whole section to its description.

Parametric surfaces

Description

The parametric representation of a 3D surface defines each point $S(\eta, \mu)$ of the surface by the following equation:

$$S(\eta, \mu) = \begin{bmatrix} f_x(\eta, \mu) \\ f_y(\eta, \mu) \\ f_z(\eta, \mu) \end{bmatrix} \tag{1.3}$$

η and μ are the two *parametric* variables, and f_x, f_y and f_z are functions from $\mathbb{R}^2$ in $\mathbb{R}$.

Parametric models represent a large family of surfaces. The most common are the Bézier surfaces and their extensions, B-spline and NURBS

(Non-Uniform Rational B-Spline) surfaces, which are particularly popular in CAD and CAM.

Pierre Bézier introduced parametric surfaces in the field of CAD in 1972 (Bézier 1972). Bézier curves and surfaces are characterized by a set of control points, which defines an $n \times m$ grid. The linear combination of these control points, following the two directions η and μ, provides all the surface points $S(\eta, \mu)$:

$$S(\eta, \mu) = \sum_{i=1}^{n} \sum_{j=1}^{m} P_{i,j} B_i^n(\eta) B_j^m(\mu) \tag{1.4}$$

where $P_{i,j}$ are the control points, and $B_i^k(t) = C_k^i t^i (1-t)^{k-i}$ the Bernstein polynomials (with $t \in [0, 1]$), which represent the blending functions. Figure 1.5 illustrates a bicubic Bézier surface, defined on a 4×4 grid of control points. Although they have strong continuity properties, these surfaces have two major drawbacks:

- No local control is possible, i.e. the displacement of a control point involves modifications on the whole surface.
- The polynomial degree increases with the number of control points. Thus it is computationally expensive to handle complex control polyhedra.

The necessity to eliminate these drawbacks has led to the creation of a new model: B-spline curves and surfaces. The formulation is quite similar to Bézier surfaces, with different blending functions. Bernstein polynomials are

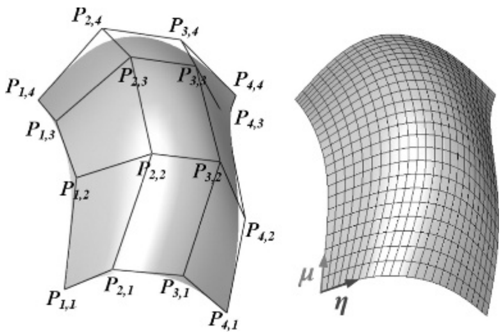


Figure 1.5 Bicubic Bézier surface (16 control points) and isoparametric curves

replaced by the B-spline functions. Considering k and l ($k \leq n$ and $l \leq m$), the respective degrees of the surface in the two directions η and μ, the formulation is:

$$S(\eta, \mu) = \sum_{i=1}^{n} \sum_{j=1}^{m} P_{i,j} N_i^k(\eta) N_j^l(\mu) \tag{1.5}$$

$N_i^k(t)$ is the B-spline basis function, of degree k, recursively defined by:

$$N_i^k(t) = \frac{(t - t_i) N_i^{k-1}(t)}{t_{i+k} - t_i} + \frac{(t_{i+k+1} - t) N_{i+1}^{k-1}(t)}{t_{i+k+1} - t_{i+1}} \tag{1.6}$$

with:

$$N_i^0(t) = 1 \text{ if } t \in [t_i, t_{i+1}[\tag{1.7}$$

$$N_i^0(t) = 0 \text{ otherwise} \tag{1.8}$$

A B-spline surface is thus a continuous, piecewise polynomial surface defined like the union of surface patches of fixed degrees (k and l) connected according to the blending B-spline functions. Vectors $t_0, t_1, \ldots, t_{n+k+1}$ and $t_0, t_1, \ldots, t_{m+l+1}$ are called the node vectors and form monotonous increasing floating values lists which represent the positions of the beginning and end of the compact supports of the blending functions. In other words, they define which proportions of each patch are present in the B-spline surface. If the nodes are uniformly spaced, the B-spline is said to be *uniform*. Figure 1.6 details this mechanism for 2D curves: a B-spline segment (analogous to a surface *patch*) is illustrated on the left while a B-spline curve made up of three B-spline segments is shown on the right.

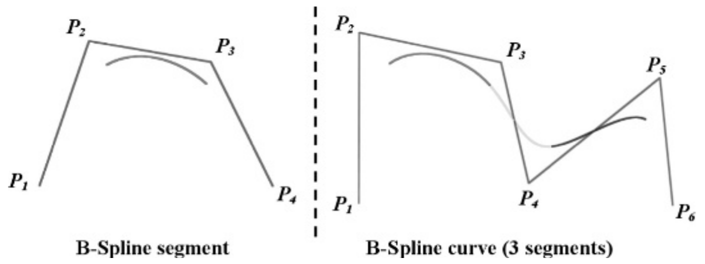


Figure 1.6 B-spline curves

B-spline curves and surfaces have many benefits, compared with Bézier ones. They permit local control: the displacement of a control point involves modifications on a delimited region of the surface (depending on degrees k and l). Moreover, the complexity and the size of the control polyhedron do not influence the degree and thus the calculation complexity of the polynomials. Finally k and l control the order of B-spline functions, therefore for the same set of control points, the resulting surface can more or less match the control polyhedron.

The next logical step is the NURBS model. These surfaces are a generalization of B-splines. A weight w_{ij} is associated with each control point. This coefficient is a tension parameter: increasing the weight of a control point pulls the surface toward that control point. NURBS models enable the definition of not only natural quadrics (such as spheres, cylinders, etc.) but also surfaces having sharp edges. For more details on parametric surfaces, interested readers are invited to consult the book by Farin (1993).

Evaluation

The representational power of parametric surfaces is really strong, especially in the case of NURBS. Further, this model has been included in the VRML (dedicated to the Internet) and MPEG-4 (2002) standards and in various 3D CAD exchange formats (IGES, STEP, etc.).

The benefits of these surfaces are numerous: they are mathematically complete and therefore easy to sample or to digitize into voxels, triangles, etc. Moreover, they do not depend on a scale factor (unlike discrete models like meshes or voxels). Lastly, they have strong continuity properties (particularly useful for design), and are much more compact in terms of amount of data than the polygonal mesh model.

However, parametric models are not as widely used as polygonal meshes, for various reasons: they remain quite complex to manipulate, and the algorithms of collision detection and intersection are quite difficult to implement. Finding intersection points between two NURBS (or B-spline) surfaces amounts to solving a complex mathematical problem (Krishnan and Manocha 1997) whereas this operation is commonplace for polygonal meshes. Also, these surfaces are defined on parametric grids, which makes them difficult to fit to an object with arbitrary topology. Thus, a whole object is often composed of several NURBS patches, associated with trimming curves which do not yet have a real standardization and which increase the volume of data.

Implicit surfaces

Description

Consider an implicit surface S as the zero set of a function f from $\mathbb{R}^3$ in $\mathbb{R}$. The set of points $P = (x, y, z)$ of the surface S defined by the function f verifies:

$$f(x, y, z) = 0 \tag{1.9}$$

Common shapes are often defined in such manner, for instance a straight line $ax + by + cz + d = 0$ or a sphere $x^2 + y^2 + z^2 - r^2 = 0$.

Thanks to this formulation, the main property of this model is to give directly the relation between the surface and every point of the 3D space. Hence, this model defines not only a surface but also a solid; indeed the sign of the function f specifies a whole partition of the 3D space. A point $P = (x, y, z)$ is such that:

- $f(x, y, z) > 0$ is outside the implicit surface;
- $f(x, y, z) < 0$ is inside the implicit surface;
- $f(x, y, z) = 0$ is on the implicit surface.

Implicit surfaces can be classified into two principal categories: *algebraic* and *non-algebraic*. A surface is algebraic if f is a polynomial. Figure 1.7 depicts such a surface. Superquadrics and hyperquadrics belong to this class, and are described in section 1.1.4.

Since algebraic surfaces are quite complex to manipulate, another formulation (non-algebraic) for implicit surfaces has been developed. The main idea

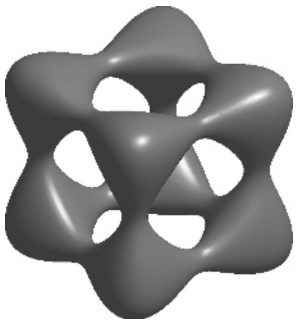


Figure 1.7 Implicit surface associated to the algebraic equation $x^4 - 5x^2 + y^4 - 5y^2 + z^4 - 5z^2 + 11.8 = 0$

is to represent an object by a set of particles. Each particle is represented by its center (a point of the 3D space) and a potential function $\Phi(r)$ decreasing with the distance r to the center. If the scene is composed with n particles, the value of the global potential function Φ^g for a point P of the 3D space is defined by:

$$\Phi^g = \sum_{i=1}^{n} \Phi_i(r_i) \tag{1.10}$$

with r_i the distance from P to the center of the ith particle. The summation of the potential functions is a sort of generalization of union operators and enables the blending of different particles. Corresponding isosurfaces (surfaces corresponding to a constant value for Φ^g) then allow modeling of arbitrary (even complex) shapes.

The first author to propose such a model was Blinn (1982), who introduced the *blob* objects defined by a Gaussian potential function $\Phi(r) = be^{-ar^2}$. Figure 1.8 illustrates the blending of two *blobs* associated with different distances.

The main drawback of Blinn's model is its *global* definition; indeed the potential function Φ is not bounded. Thus other potential functions have been introduced (Murakami and Ichihara 1986; Nishimura *et al.* 1985; Wyvill *et al.* 1986) which permit bounding of the *blob* influence in order to speed up calculations.

This model was improved and generalized by replacing the center of the particles by some more complex primitives like segments or faces, in order to create *skeleton-based* implicit surfaces. The potential field value generated by an element of the skeleton, for a point P of the 3D space, is then defined by a potential function $\Phi(r)$, with r the distance from P to the

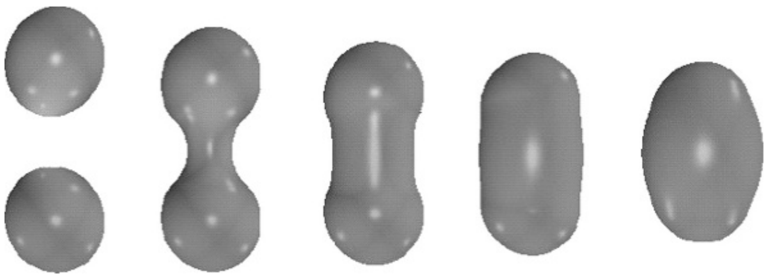


Figure 1.8 Unions of two *blobs* at different distances

skeleton (Bloomenthal and Wyvill 1990). However, this modeling technique can produce artifacts or discontinuities, particularly for a non-convex skeleton (when the radius of curvature of the skeleton is smaller than the radius of influence of the potential function, for instance). Thus a new class of skeleton-based implicit surfaces was introduced, based on the principle of convolution (Bloomenthal and Shoemake 1991). These surfaces are now defined by using integrals of the potential field along the skeleton elements. Numerous authors are developing more and more complex and effective implicit models: Wyvill *et al.* (1999) introduced the *BlobTree* which organizes the elements of the skeleton in a hierarchical graph structure, in which nodes represent blending operations, boolean operations or deformations. This model was extended by Barbier *et al.* (2005) who introduced levels of detail. In the same way, Angelidis and Cani (2002) and Hornus *et al.* (2003) presented a multiresolution implicit model, based on convolution along a skeleton represented by subdivision curves.

Evaluation

Through their different representations (algebraic, particles, skeletons), implicit surfaces offer real modeling power from the more intuitive (*blobs*, skeletons) to the more *mathematical* (algebraic surfaces). Arbitrary topologies can also be modeled (provided that there is no boundary or sharp edge). Moreover, these surfaces make the algorithms of intersection, belonging or blending particularly simple. Deformation and animation algorithms are also easily facilitated, hence implicit surfaces are especially used in the medical field, for the modeling of body parts. Further this model is particularly compact. In the case of a sphere, for example, the implicit formulation is quite simple: $(x - c_x)^2 + (y - c_y)^2 + (z - c_z)^2 - r^2 = 0$.

However, sampling and distance calculations are quite difficult to handle. Contrary to the parametric models, points of the implicit surfaces are not directly calculable. Thus, triangulate surfaces require quite complex algorithms like the *marching cubes* (Lorensen and Cline 1987) or the more recent *marching triangles* (Akkouche and Galin 2001). In spite of a relatively high representational power, due particularly to the recent non-algebraic implicit modeling techniques (Angelidis and Cani 2002; Barbier *et al.* 2005; Bloomenthal and Shoemake 1991; Bloomenthal and Wyvill 1990; Hornus *et al.* 2003; Muraki 1991), implicit surfaces are especially adapted for organic modeling, and cannot represent mechanical or CAD objects because of the difficulty of representing planes or sharp edges. Concerning algebraic surfaces, there is no possible local control, or local deformations.

Subdivision surfaces

Description

The basic idea of subdivision is to define a smooth shape from a coarse polyhedron by repeatedly and infinitely adding new vertices and edges according to certain subdivision rules. Figure 1.9 illustrates this subdivision mechanism for a 2D curve. Figure 1.9(a) shows four points connected by segments (the control polygon), while Figure 1.9(b) shows the polygonal curve after a refinement step: three points have been added and the old points have been displaced. By repeating this refinement several times, the polygonal curve takes on the appearance of a smooth curve. Continuity properties of the limit curve depend on the subdivision rules. Similarly, an example of a subdivision surface is presented in Figure 1.10. At each iteration, each quadrangle is divided into four, new points are thus inserted and the old ones are displaced. After several subdivision iterations, the surface seems smooth (see Figure 1.10(d)).

The main difficulty is to find subdivision rules which lead to certain good properties such as calculation simplicity, local control, continuity and pleasant visual aspect. In the case of the subdivision curve presented in Figure 1.9, the polygonal curve is firstly linearly subdivided (i.e. new points are added in the middle of the segments), and then each point P_i is replaced by a linear combination of itself and its direct neighbors P_{i-1} and P_{i+1}, following the smoothing mask:

$$P_i' = \frac{1}{4}(P_{i-1}' + 2P_i + P_{i+1}') \tag{1.11}$$

With these rules, the limit curve corresponds to a uniform cubic B-spline (see Figure 1.9).

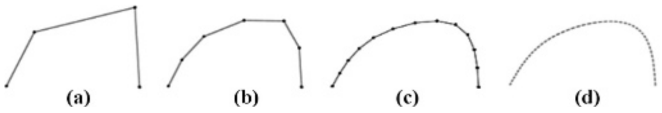


Figure 1.9 Example of a subdivision curve: (a) control polygon, (b, c) two refinement steps, (d) limit curve

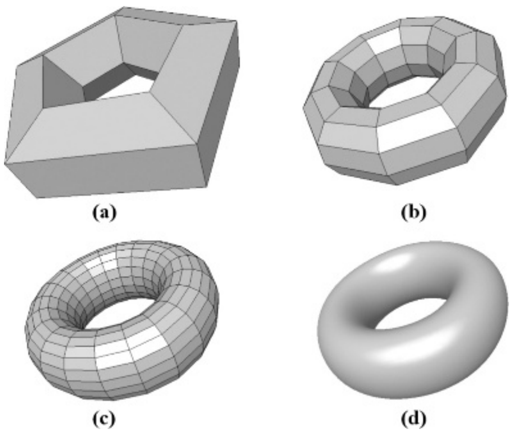


Figure 1.10 Example of a subdivision surface: (a) control polyhedron, (b, c) two refinement steps, (d) limit surface

Doo and Sabin (1978) and Catmull and Clark (1978) (see Figure 1.10) were the first authors to introduce subdivision surface rules. Their schemes respectively generalized biquadratic and bicubic tensor product B-splines (Farin 1993). Today, many subdivision schemes have been developed, based on quadrilateral (Kobbelt 1996; Peters and Reif 1997) or triangular meshes (Loop 1987; Zorin *et al.* 1996). Recently, Stam and Loop (2003) and Schaefer and Warren (2005) have introduced subdivision schemes for mixed quad/ triangle polyhedra. Moreover, special rules have been introduced by Hoppe *et al.* (1994) to handle sharp edges (i.e. to preserve the sharpness of a given control edge). A subdivision scheme can be described by:

- **A topological component**: Every subdivision scheme changes the connectivity of the polyhedron, by adding/removing vertices or flipping edges. Then we can classify them into two main categories: *primal* schemes which do not remove old vertices, and *dual* schemes which remove them.
- **A geometric component**: The displacement of the vertices can be seen as a smoothing operation on the input polyhedron. Considering this operation, subdivision schemes can also be split into two main classes: *interpolating* schemes which do not modify the position of the old vertices, and *non-interpolating* schemes which change the position of both old and new vertices.

Table 1.1 Classification of principal subdivision schemes

Primal	Triangles	Quadrangles	***Dual*** (quadrangles)
Non-interpolating	*Loop* (C^2) (Loop 1987)	*Catmull–Clark* (C^2) (Catmull and Clark 1978)	*Doo–Sabin* (C^1) (Doo and Sabin 1978) *Midedge* (C^1) (Peters and Reif 1997)
Interpolating	*Butterfly* (C^1) (Zorin *et al.* 1996)	*Kobbelt* (C^1) (Kobbelt 1996)	

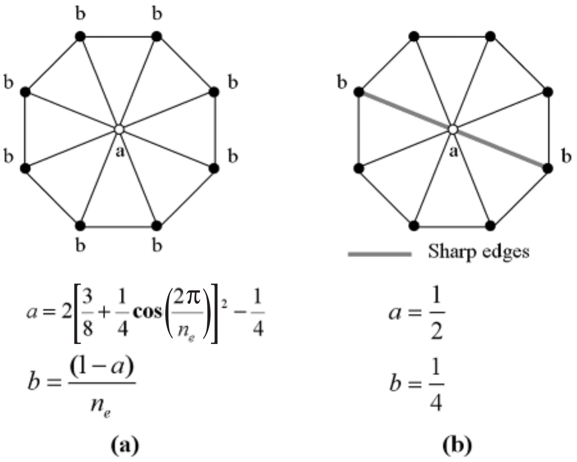

b
a
Sharp edges
$a = 2\left[\frac{3}{8} + \frac{1}{4}\cos\left(\frac{2\pi}{n_e}\right)\right]^2 - \frac{1}{4}$
$b = \frac{(1-a)}{n_e}$
(a)
$a = \frac{1}{2}$
$b = \frac{1}{4}$
(b)

Figure 1.11 Smoothing masks for Loop subdivision rules: (a) standard vertex, (b) *sharp* vertex

Hence, we can classify existing subdivision schemes. This classification is detailed in Table 1.1.

A subdivision scheme is generally described by smoothing masks. Figure 1.11 illustrates the different masks associated with the Loop (Loop 1987) subdivision rules which apply on triangular meshes. We consider the subdivision scheme as being composed of two distinct steps: a linear subdivision (a vertex is added at the middle of each edge), and then a smoothing (each vertex is replaced by a linear combination of itself and its direct neighbors). Smoothing coefficients are presented in Figure 1.11(a) with n_e the valence of the considered vertex. Special rules were introduced by Hoppe *et al.* (1994) for handling sharp and boundary edges. Figure 1.11(b) illustrates the smoothing coefficients for vertices shared by two sharp edges.

For more details about subdivision surfaces, the reader may refer to the book by Warren and Weimer (2002).

Evaluation

Subdivision surfaces offer many benefits. Firstly they can be generated from arbitrary meshes (arbitrary topology), which implies no need for trimming curves (which are necessary for NURBS surfaces, defined on parametric rectangles). Secondly, the majority of existing subdivision schemes are really easy to implement and linear in complexity. Further, they can be generated at any level of detail, by iterative subdivision of the control mesh, according to the terminal capacity for instance. Local control is possible (which is not the case for algebraic implicit surfaces) with the possibility of creating and preserving sharp features. Most of the existing schemes produce at least C^1 continuous (except around sharp edges, of course) limit surfaces. Lastly, it is an extremely compact model since a smooth surface is represented by a coarse control mesh (thus no need to encode node vectors or trimming curves as for the NURBS). For all these reasons, subdivision surfaces are now widely used in computer graphics and 3D imaging, and particularly in the fields of animation, reconstruction and CAD, and have been integrated into the MPEG-4 standard (MPEG-4 2002).

One of the main drawbacks of subdivision is the lack of parametric formulations. Thus they cannot be directly evaluated at any points. Moreover, the control mesh has to be adapted to the considered rules (based on quads or triangles). Also, for certain subdivision schemes, the continuity properties are not as strong as for parametric surfaces, particularly around *extraordinary* points (vertices of the mesh having a non-regular valence).

1.1.4 Volumetric Representation

Primitive-based models

These models decompose a complex 3D object into a set of simple ones, called primitives, which are then combined using different operations. These primitives are generally arranged in a graph that allows a structural modeling of the object, particularly useful for indexing and recognition purposes.

Several types of primitives exist: generalized cylinders (Binford 1971; Ponce *et al.* 1989; Zeroug and Nevatia 1996), geons (Biederman 1985; Nguyen and Levine 1996), superquadrics (Barr 1981), supershapes (Gielis *et al.* 2003), hyperquadrics (Han *et al.* 1993) and other implicit polynomial models (Keren *et al.* 1994). These models can be classified in two main categories: *quantitative* models having a real representational power (superquadrics, hyperquadrics, implicit polynomials) and *qualitative* models which have a symbolic and structural representation purpose (geons, generalized cylinders). The objective of this chapter is to present true modeling techniques, thus we will only detail quantitative models and more particularly superquadric models and their extensions of supershapes and hyperquadrics, implicit polynomial models having already been detailed in Section 1.1.3.

Superquadrics

Superquadrics were introduced by Barr (1981) in the early 1980s. They are an extension of quadrics: two additional parameters were introduced, ε_1 and ε_2, which control the latitudinal and longitudinal curvatures. These primitives have the specificity to possess both implicit and parametric formulations. The most commonly used ones are superellipsoids, defined as follows.

Parametric formulation:

$$S(\eta,\mu)=\begin{bmatrix} a_1\cos^{\varepsilon_1}(\eta)\cos^{\varepsilon_2}(\mu)\\ a_2\cos^{\varepsilon_1}(\eta)\sin^{\varepsilon_2}(\mu)\\ a_3\sin^{\varepsilon_1}(\eta)\end{bmatrix};\ -\frac{\pi}{2}\le\eta\le\frac{\pi}{2},\ -\pi\le\mu\le\pi \quad (1.12)$$

Implicit formulation:

$$F(x,y,z)=1-\left[\left(\frac{x}{a_1}\right)^{\frac{2}{\varepsilon_2}}+\left(\frac{y}{a_2}\right)^{\frac{2}{\varepsilon_2}}\right]^{\frac{\varepsilon_2}{\varepsilon_1}}+\left(\frac{z}{a_3}\right)^{\frac{2}{\varepsilon_1}} \quad (1.13)$$

Parameters a_1, a_2, a_3 control the size of the superellipsoid while ε_1 and ε_2 control its curvature. Figure 1.12 illustrates some examples of superellipsoids associated to different ε_1 and ε_2 values. Hence, with only five parameters, this model yields the representation of a quite broad range of shapes such as cubes, cylinders, octahedra or ellipsoids. Barr (1981) has also defined two other classes of superquadric, namely supertoroid and superhyperboloid, with one or two sheets, but their use remains marginal.

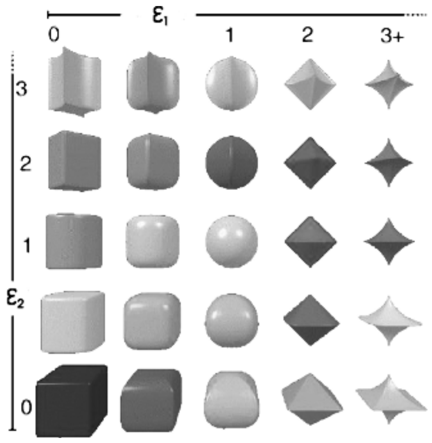


Figure 1.12 Superellipsoid examples and associated ε_1 and ε_2 values. Data Courtesy of L Chevalier

Superquadrics, and more particularly superellipsoids, have a high representational power relative to their small number of parameters; the model is extremely compact. The fact of having both implicit and parametric formulations is also quite interesting, and enables the model to be adapted to the application. Indeed, intersection or blending algorithms are easy to handle using the implicit equation while the parametric formula facilitates sampling or meshing. These surfaces are quite popular in computer graphics (Jaklic and Solina 2003; Jaklic *et al.* 2000; Montiel *et al.* 1998).

In spite of a rather high representational power considering the extremely small number of parameters, it still remains much lower than those of surface-based models (parametric, implicit, subdivision). Indeed it is quite difficult to perfectly model an arbitrary object even by considering a whole graph of superquadrics. Thus, many authors have introduced deformation techniques in order to increase the flexibility of this model, and particularly to reduce its heavy symmetry constraints.

Barr (1984) proposes a set of global deformations: torsion, folding, etc. Terzopoulos and Metaxas (1991) add local deformation coefficients. Bardinet *et al.* (1994) consider control-point-based deformations and Zhou and Kambhamettu (2001) replace the exponents ε_1 and ε_2 by functions depending on latitudinal and longitudinal angles. Blanc and Schlick (1996) introduce a new model, ratioquadrics, in which the exponential functions are replaced by numerically simpler ones. Lastly, DeCarlo and Metaxas (1998) propose blending operations between superquadrics.

Supershapes

Supershapes were recently introduced by Gielis *et al.* (2003). Supershapes can be considered as a generalization of superquadrics, allowing the modeling of asymmetrical shapes. Like superquadrics, they have both parametric and implicit formulations.

Parametric formulation:

$$S(\eta,\mu)=\begin{bmatrix} r_1(\mu)r_2(\eta)\cos(\eta)\cos(\mu) \\ r_1(\mu)r_2(\eta)\cos(\mu)\sin(\mu) \\ r_2(\eta)\sin(\eta) \end{bmatrix};-\frac{\pi}{2}\leq\eta\leq\frac{\pi}{2},-\pi\leq\mu\leq\pi \tag{1.14}$$

Implicit formulation:

$$F(x,y,z)=1-\frac{x^2+y^2+r_1^2(\mu)z^2}{r_1^2(\mu)z^2r_2^2(\eta)z^2} \tag{1.15}$$

with

$$r(\mu)=\frac{1}{\sqrt[n_1]{\frac{1}{a}\cos\left(\frac{m\mu}{4}\right)^{n_2}+\frac{1}{b}\sin\left(\frac{m\mu}{4}\right)^{n_3}}} \tag{1.16}$$

n_1, n_2 and n_3 are three shape coefficients and the term $m\mu/4$ allows a rational or irrational number of symmetry, while a and b control the size of the shape. Figure 1.13 illustrates some examples of supershapes.

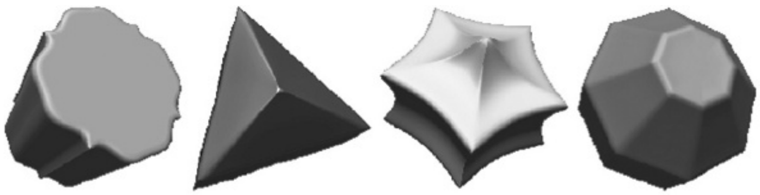


Figure 1.13 Supershape examples. (From the work of Fougerolle *et al.* (2006)). Reproduced with kind permission of Springer Science and Business Media

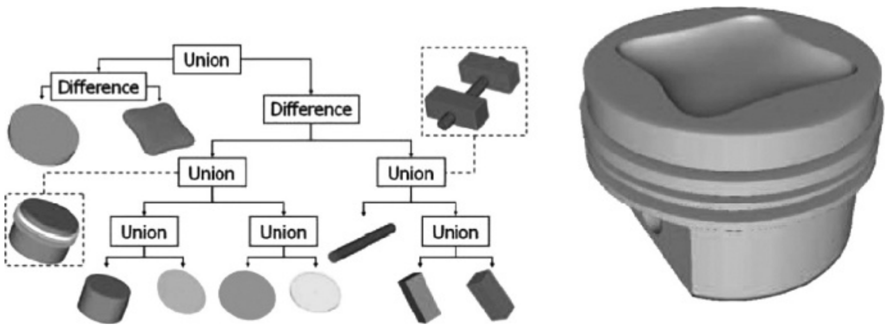


Figure 1.14 Combination of several deformed supershapes (left) and corresponding constructed mesh (right) by the algorithm of Fougerolle *et al.* (2005). Reproduced by permission of IEEE

This model is very recent and exhibits a higher representational power than superquadrics (by breaking the symmetry constraints) while keeping the same benefits of both implicit and parametric formulations. Several authors have investigated the use of combinations of deformed supershapes to represent more and more complex 3D models. Figure 1.14 illustrates a mechanical model composed with 9 supershapes associated with boolean operations. Fougerolle *et al.* (2005) introduced a recent algorithm to accurately polygonize such supershape combinations, and proposed a new formulation for the implicit representation (Fougerolle *et al.* 2006).

Hyperquadrics

Hyperquadrics (Vaerman *et al.* 1997) are a generalization of superquadrics, they enable the representation of a higher variety of shapes and do not have symmetry constraints. However, they are more complex to handle and, contrary to superquadrics, they only have an implicit formulation (Fougerolle *et al.* 2006):

$$\sum_i S_i + e^{\sum_j S_j} = 0 \tag{1.17}$$

where S is a superquadric equation.

Although hyperquadrics have a higher representational power than superquadrics, their use remains marginal. Their high complexity and the lack of parametric formulations make them of limited interest. Like superquadrics,

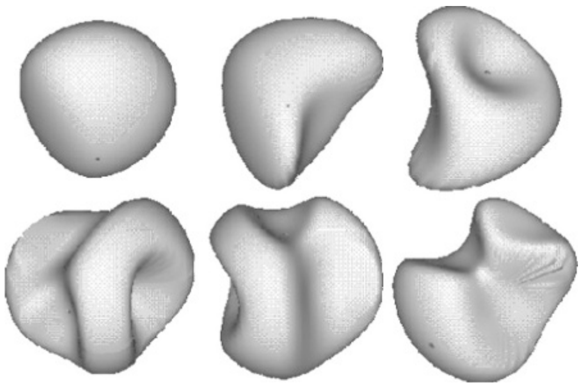


Figure 1.15 Hybrid hyperquadrics, introduced by Cohen and Cohen (1996). Reprinted from Cohen and Cohen, 1996, with permission from Elsevier

these primitives are used especially for organic part reconstruction and simulation in the medical field. The difficulty in controlling the shape has led researchers to introduce local deformations; thus, Cohen and Cohen (1996) introduced the hybrid hyperquadrics, by adding an exponential term to the equation. Examples of their model are illustrated in Figure 1.15.

Constructive solid geometry (CSG)

Description

This model represents a 3D object by a set of linear transformations and boolean operations (union, intersection, difference) applied on elementary primitives (sphere, torus, cone, cubes). Thus the object is represented by a tree (called a *constructive tree*), with a primitive associated with each leaf and an operator associated with each non-terminal node. Figure 1.16 illustrates such a model.

Evaluation

Constructive solid geometry is only used in CAD to represent mechanical parts. The model is extremely compact and permits representation of the whole modeling process. However, this model is very hard to display, and involves discontinuity problems at the joints between primitives. Moreover, there is no associated standard format, and from one piece of CAD software to

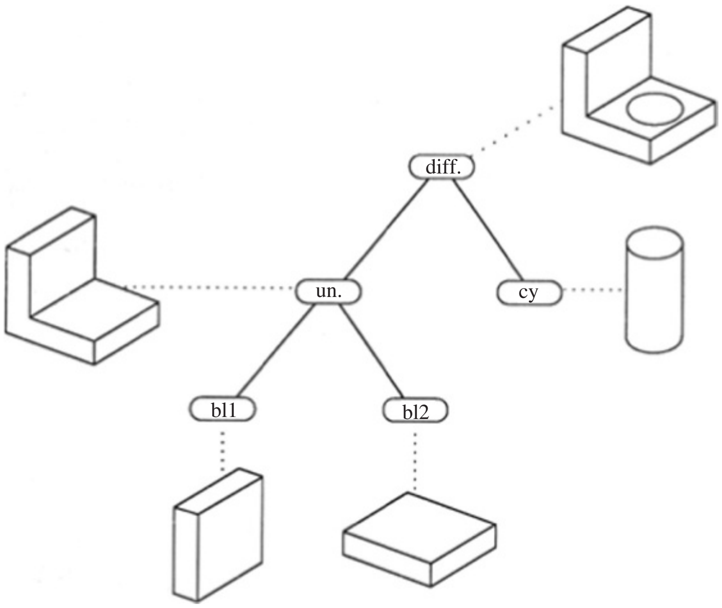


Figure 1.16 Constructive representation of a 3D object

another, operators and primitives are not the same. Further, the geometrical and physical properties are quite complex to calculate. Lastly, the inverse problem of constructing a CSG tree from a set of 3D data (3D point cloud, or 3D mesh) is almost impossible to handle.

Discrete model

Description

This model decomposes the 3D space into volume elements (called voxels). Thus a 3D object is represented by the set of voxels constituting (or intersecting) its volume. An example of a discrete 3D object is illustrated in Figure 1.17.

Evaluation

This discrete model is quite simple to construct and facilitates boolean operations and calculation of physical properties. Moreover, several authors have

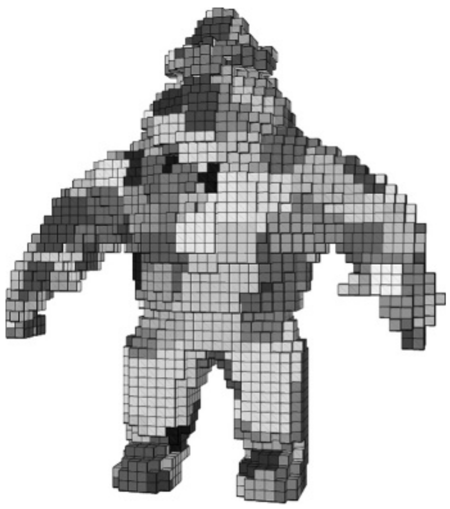


Figure 1.17 Discrete representation of the 3D object *Al*. Courtesy of I Sivignon

generalized differential operators to such models (Coeurjolly and Klette 2004; Flin *et al.* 2005). This representation is used particularly in the medical field, since it comes directly from tomographic reconstruction.

However, this model is very expensive in terms of memory and gives only an approximation of the 3D object (arising from its discrete nature) according to a certain level of precision. Moreover, algorithms of ray tracing, meshing or surface or planes extraction are quite complex problems (Kobbelt *et al.* 2001; Sivignon *et al.* 2004). Figure 1.17 illustrates digital planes retrieved using the algorithm from Sivignon *et al.* (2004) (where a color is used to represent a plane).

1.2 3D Data Source

There are several ways to produce 3D content, but basically we can distinguish two principal classes:

- The *acquisition*, which reproduces a real-world object, like a camera for 2D images.
- The *manual creation*, using specific software, similar to a painting of 2D images.

1.2.1 Acquisition

The main objective is to create a 3D object which reproduces a given real-world object, like a photograph represents a real-world image. There are mainly two classes of technologies: those which analyse and reconstruct the *surface* of the object (e.g. 3D scanner) and those which reconstruct its *volume* (e.g. tomography).

3D scanner

A 3D scanner works like a camera, but instead of collecting color information about a surface, it collects *distance* information. Basically, each scan produces a picture where each pixel contains the distance to the surface; this picture is called a depth (or *range*) image. Usually to analyze properly a complex 3D object, multiple scans (i.e. range images) from different directions are necessary. These *range images* are then merged together to produce a cloud of points representing the surface (see Figure 1.1(a)). This operation is called *registration* or *alignment* (Salvi *et al*. 2007) and consists of bringing the range images, coming from different directions, within the same 3D coordinate system. Figure 1.18 illustrates examples of registration.

There are several types of 3D scanners:

- *Laser scanners* emit a beam toward the object and detect its reflection. There are basically two technologies. The *time-of-flight* (or *laser range-finder*) scanner measures the time taken by the beam to be reflected by the target and returned to the sender and then deduces the distance according to the speed of light. This technology is not quite precise but can operate

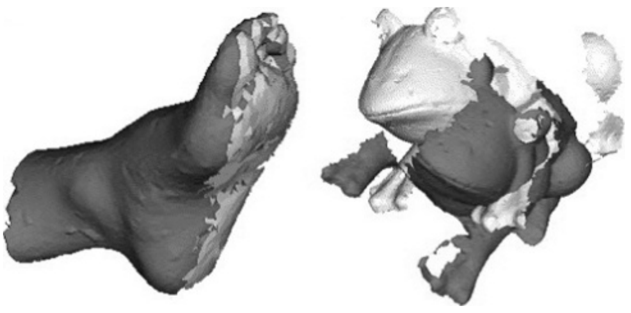


Figure 1.18 Examples of registration of two range images: correct (left) and incorrect (right). Reproduced from Salvi *et al*. 2007.

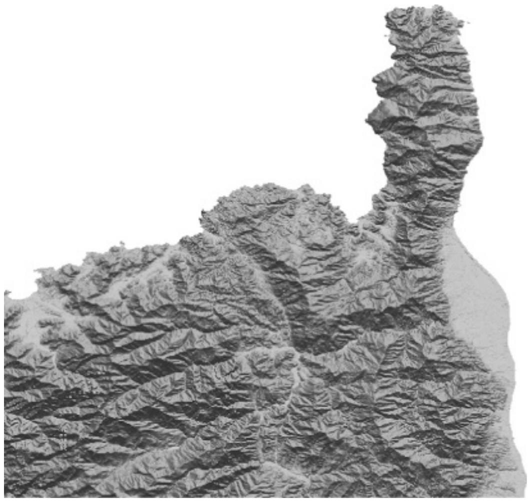


Figure 1.19 Triangular mesh of the north of Corsica, reconstructed according to a digital elevation model. Courtesy of DISI by the AIM@SHAPE Shape repository

over very long distances and thus is particularly suitable for measuring large scenes. For example, with this technique, the *NASA Shuttle Radar Topographic Mission* provides digital elevation models for over 80 % of the globe; Figure 1.19 illustrates a triangular mesh of the north of Corsica, reconstructed according to a digital elevation model. The other technology is the *triangulation* scanner which emits a beam toward the object and uses a camera to analyze the location of the laser dot on the object. The laser dot, the camera and the laser emitter form a triangle which locates very precisely the corresponding 3D point (accuracy of several micrometers). However, such a technique is limited to a depth of several meters. Figure 1.20 illustrates the laser scanning of the head of Michelangelo's David and the corresponding 3D model, from the Digital Michelangelo Project (Levoy *et al*. 2000).

- *Contact scanners* (or *coordinate measuring machines*) use a small arm (automatic or manual) to touch the 3D object and record the corresponding 3D points. However, such a technique can damage the object (because of the contact) and is much slower than laser scanners.

There are also some systems that are able to digitize a 3D model without laser scanning or contact, but only with several cameras using stereo reconstruction techniques. Hence two images of the same object, associated with adapted

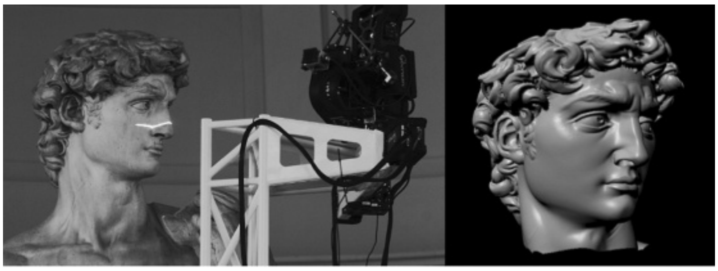


Figure 1.20 Scanning of the head of Michelangelo's David with a triangulation laser scanner (left) and corresponding 3D model (right). Reproduced from the Digital Michelangelo Project (Levoy *et al*. 2000), Stanford University, Copyright © Marc Levoy – Paul Debevec

lighting conditions, are able to reconstruct the 3D geometry, under some shape constraints.

Scanning technology is particularly used for:

- Cultural heritage (see Figure 1.20), to preserve in digital form historical pieces or sites.
- Reverse engineering, which basically consists of scanning a mechanical part for instance and then reconstructing the corresponding CAD model (typically a set of parametric surfaces).
- Geographic information science, where the objective is to model 3D geographical information, like terrain models (see Figure 1.19), for applications in cartography or spatial planning.

Tomography

Tomography is a technique that consists of reconstructing the volume of a 3D object from a set of external measures which often involve emitting a certain signal through the object toward a sensor and analyzing the response. The nature of the retrieved volumetric data depends on the nature of the signal and of the sensors. For instance, the *computed axial tomography scanner* (or CT scan) measures the ability of the material to absorb X-rays; this scanner sends X-ray beams through the object, and a sensor measures their residual intensity. Similarly *echography* measures the acoustic impedance of a material by measuring the attenuation of ultrasound sent through the object. Another example is the *nuclear magnetic resonance imaging* (MRI) scanner which measures the

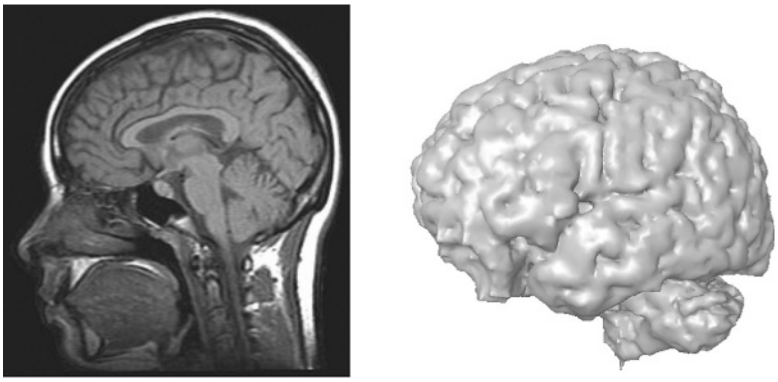


Figure 1.21 A 3D magnetic resonance image of the head, vertical cross-section, and the result after segmentation. The brain 3D model is Courtesy of INRIA by the AIM@SHAPE Shape repository

proton density and structure of a material. The object is placed in a powerful magnetic field. The hydrogen nuclei are excited and then relax by emitting an electromagnetic pulse which is measured by a sensor. Lastly, *positron emission tomography* (PET) works by measuring the radioactive emission of a tracer injected in the object (or the body) for analysis.

The tomography technology produces an image of the considered physical measures (X-ray absorption, ultra sound response, etc.) which has to be converted to 3D data by a mathematical *reconstruction* process. Unlike 3D laser scanners, the tomography technology does not provide a cloud of points but a volumetric model in the form of a 3D grid where each element contains a kind of density value or other information. To be properly utilized, these data often have to be segmented to obtain a 3D binary grid or a polygonal or tetrahedral mesh representing a specific part. Figure 1.21 illustrates a section of a 3D MRI image and the segmentation result.

Tomography technologies mostly concern medical imagery, where analyzing the interior of an organ is critical to detect eventual diseases or pilot surgical operations. Another application of tomography is in geophysics to analyze some aspects of the Earth.

1.2.2 Manual Creation

A 3D content can be obtained by reproducing a real-world object (*acquisition*, see previous section), but can also be created manually by designers using

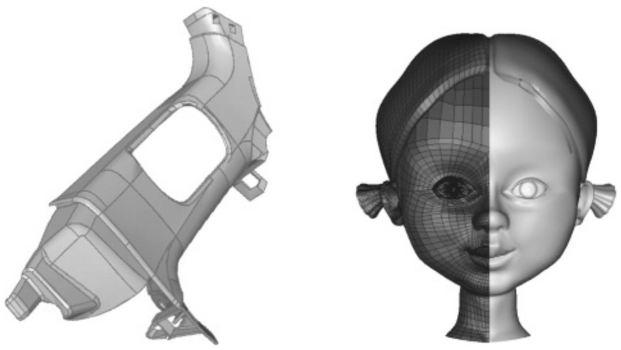


Figure 1.22 Examples of hand-designed 3D objects. Left: CAD object from a Laguna car. Right: Little girl from a 3D modeler (Software Amira)

specific software. Their are two principal types of software: *CAD software* and 3D *modelers*.

CAD software

CAD software helps engineers or architects to design industrial objects, machinery, manufactured content, buildings, etc. This software is based mostly on parametric surfaces and can produce 3D content to a very high precision since this content is then used for manufacturing the real object. An example of a CAD model composed with parametric surfaces is given in Figure 1.22 (left); the boundaries between parametric surfaces are indicated by the solid lines.

3D modelers

Whereas CAD software aims at designing high-precision technical objects, 3D modelers are rather adapted to produce artistic 3D content and 3D animations. The main applications are videogames and 3D movies. Most 3D modelers are based on a polygonal mesh representation since this model is highly adapted for rendering or simple operations like collision detection etc. Texture and color information are often added to the model (see Figure 1.22, right).

1.2.3 3D Object Databases

A critical issue in the design of new algorithms and evaluation of their performance is the presence of open 3D object databases. However, only a few 3D databases exist presently. Recently, the AIM@SHAPE Shape Repository was created within the Network of Excellence project AIM@SHAPE (Advanced and Innovative Models And Tools for the development of Semantic-based systems for Handling, Acquiring, and Processing knowledge Embedded in multidimensional digital objects, 2004–2007). This repository contains a collection of digitally scanned 3D shapes (3D meshes, range images, volumetric models) and offers some interesting functionalities like multiresolution formats. Another example is the Princeton Shape Benchmark which was created specifically to compare 3D indexing methods, hence the objects are classified into several classes. Lastly, some other databases exist for specific 3D face applications: namely, the $3D_RMA$ database at the Signal and Image Center (SIC) at the Royal Military Academy of Belgium, and the database of the University of Notre Dame (UND) in the United States.

1.3 3D Quality Concepts

Compression, or even watermarking, can introduce degradations of the visual quality of a 3D object. Since the ultimate application of these processes is often to be seen by human beings, the need for efficient tools to measure the loss of quality or the visual differences between 3D objects is critical.

Some geometric measures exist between 3D objects, such as the *Hausdorff* distance, where $e(p, M)$ represents the distance from a point p of the 3D space to a 3D object M:

$$e(p, M) = \min_{\{p' \in M\}} d(p, p') \tag{1.18}$$

with d the Euclidean distance between points p and p'. Then the asymmetric Hausdorff distance between two 3D objects M and M' is:

$$H_a(M, M') = \max_{\{p \in M\}} e(p, M') \tag{1.19}$$

The symmetric Hausdorff distance is then defined as follows:

$$H_s(M, M') = \max\left\{H_a(M, M'), H_a(M', M)\right\} \tag{1.20}$$

The Hausdorff distance is based on an L_∞ norm. Other distances exist based on L_1 or L_2 norms; for example, the mean distance (L_1) is defined as follows:

$$D_{L1}(M, M') = \frac{1}{\text{Area}(M)} \int_M e(p, M')dM \tag{1.21}$$

These geometric distances are available in many software packages (Aspert *et al.* 2002; Cignoni *et al.* 1998), especially for polygonal meshes, but are not well matched with visual human perception. This *perceptual gap* was investigated a great deal in the field of image processing.

1.3.1 Existing 2D Perceptual Metrics

A review of existing 2D perceptual metrics can be found in Eckert and Bradley (1998). Basically there are two different approaches: *computational* and *ad hoc*.

Computational metrics, like the visible difference predictor (VDP) of Daly (1993), consist of complex numerical models taking into account psychophysical and physiological evidence. These models often rely on the same perceptual attributes (Eckert and Bradley 1998):

- The contrast sensitivity function (CSF) which defines the contrast at which frequency components become just visible.
- The channel decomposition, claiming that human vision consists of several channels selective to spatial frequency and to orientation.
- The masking effect which defines the fact that a signal can be masked by the presence of another signal with similar frequency or orientation.

These attributes lead to filtering operations (according to the CSF), filter bank decomposition, error normalization (masking effect) and error summation across frequency bands and space (usually using the Minkowski metric).

Ad hoc metrics consider simpler mathematical measures, intuitively relating to visual perception and/or introducing penalties for specific artifacts. A typical example is the work of Marziliano *et al.* (2004) which aims at detecting and quantifying blocking and ringing artifacts of JPEG compression.

Recently, Wang *et al.* (2004) introduced an alternative framework for image quality assessment which does not rely on a summation of kinds of perceptual errors, but on the degradation of the structural information.

1.3.2 Existing 3D Perceptual Metrics

Transposing complex *computational* metrics from 2D images to 3D objects is quite difficult, and to the best of our knowledge was only investigated by Ferwerda *et al.* (1997). They proposed a masking model, extending the

Daly VDP, which demonstrates how surface texture can mask the polygonal tessellation.

Most of the work on 3D perceptual distances has focused on three specific applications: realistic rendering, mesh simplification and evaluation of specific processes (compression or watermarking).

The objective of perceptually driven rendering is to determine, according to the location of the observer, which level of detail (LoD) to use to satisfy frame rate and image quality requirements. In fact these methods are based on 2D image perceptual models described in the previous section. Reddy (1996, 2001) analyzed the frequency content in several pre-rendered images to determine the best LoD. In a different way, Bolin and Meyer (1998) and more recently Farrugia and Peroche (2004) used perceptual models to optimize the sampling for ray tracing algorithms. Most of these works concern off line rendering. Luebke and Erikson (1997), Luebke and Hallen (2001) and Dumont *et al.* (2003) presented view-dependent simplification algorithms for real-time rendering, based on the worst cases of imperceptible contrast and spatial frequency changes. Williams *et al.* (2003) extended the work of Luebke *et al*. to textured objects. Although the existing work on realistic rendering is based mostly on 2D image metrics, several authors have considered some kinds of 3D metrics, namely Tian and AlRegib (2004) and Pan *et al.* (2005). Their metrics rely respectively on geometry and texture deviations (Tian and AlRegib 2004) and on texture and mesh resolutions (Pan *et al.* 2005).

Some real 3D metrics (*ad hoc*) are used to control mesh simplification algorithms, which consist of reducing the number of vertices while preserving the visual appearance. Kim *et al.* (2002) stated that human vision is sensitive to curvature changes and proposed a *discrete differential error metric* (DDEM), which is basically the following, between two vertices v and v':

$$\mathrm{DDEM}(v, v') = Q(v, v') + T(v, v') + C(v, v') \tag{1.22}$$

with Q a quadratic distance, T a normal vector difference and C a discrete curvature difference. In a different way, Howlett *et al*. (2005) pilot their simplification algorithm so as to emphasize visually salient features, determined by an eye tracking system. Lee *et al.* (2005) followed a similar approach but automatically extract the saliency from the input mesh by computing kinds of multiresolution curvature maps.

Recently several authors have investigated the use of perceptual metrics (*ad hoc*) for the evaluation of specific applications. Karni and Gotsman

(2000), in order to evaluate properly their compression algorithm, introduced the *geometric Laplacian* (GL), which measures the smoothness of a vertex v:

$$\mathrm{GL}(v) = v - \frac{\sum_{i \in n(v)} l_i^{-1} v_i}{\sum_{i \in n(v)} l_i^{-1}} \tag{1.23}$$

where $n(v)$ is the set of indices of the neighbors of v, and l_i the Euclidean distance from v to v_i. $\mathrm{GL}(v)$ represents the difference vector between v and its new position after a Laplacian smoothing step. Thus it represents a measure of smoothness: the lower it is, the smoother is the surface around v. Using this Geometric Laplacian, Karni and Gotsman introduced a perceptual metric between two meshes X and Y, with the same connectivity, containing n vertices:

$$\mathrm{GLD}(X, Y) = \frac{1}{2n} \left(\sum_{i=0}^{n-1} \left\| v_i^x - v_i^y \right\| + \sum_{i=0}^{n-1} \left\| \mathrm{GL}(v_i^x) - \mathrm{GL}(v_i^y) \right\| \right) \tag{1.24}$$

where v_i^x and v_i^y are the respective ith vertices from X and Y. In the same way, Drelie Gelasca *et al.* (2005) proposed a new metric based on global *roughness* variation, to measure the perceptual quality of a watermarked mesh. They define the *roughness* as the variance of the difference between a 3D model and its smoothed version, similar to the geometric Laplacian of Karni and Gotsman (2000). Corsini *et al.* (2005) presented a similar *roughness*-based measure. Rondao-Alface *et al.* (2005) presented two other metrics to benchmark watermarking schemes, one based on a measure of distortion between several 2D views, and the other based on the distortion of energy calculated using 2D parameterization of the meshes. Finally Lavoué *et al.* (2006) introduced a structural distortion measure which reflects the visual similarity between two meshes in a general-purpose context: the MSDM (3D Mesh Structural Distortion Measure).

1.3.3 A Perceptual Measure Example

The MSDM, from Lavoué *et al.* (2006), follows the concept of structural similarity recently introduced for 2D image quality assessment by Wang *et al.* (2004). This measure, which aims at reflecting the perceptual similarity between two 3D objects, is based on curvature analysis (mean, standard deviation, covariance) on local windows of the meshes. The authors

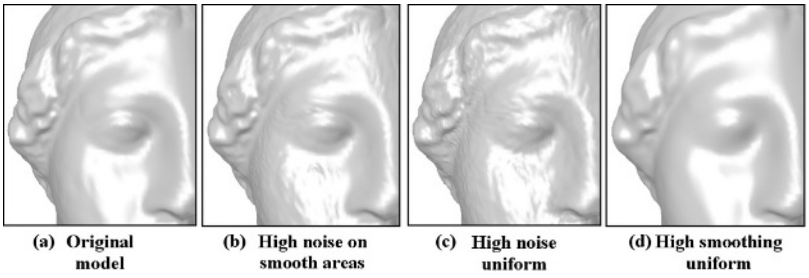


Figure 1.23 (a) Zoom on the original *Venus* object, (b) high noise (maximum deviation = 0.012) on smooth regions (MOS = 8.8, MSDM = 0.64, $D_{L1} = 0.16$), (c) high noise on the whole object (MOS = 9.4, MSDM = 0.70, $D_{L1} = 0.26$), (d) high smoothing (30 iterations) on the whole object (MOS = 8.1, MSDM = 0.58, $D_{L1} = 0.25$)

have evaluated their metric through a subjective experiment: a test corpus was created, containing distorted versions (issuing from noise addition and smoothing operations) of several 3D objects. Examples of distortions are provided in Figure 1.23 for the *Venus* 3D model. Then a pool of human observers were asked to rate the visual similarity of these versions with the corresponding original objects. Thus each object of this corpus is associated with a mean opinion score (MOS) reflecting its subjective similarity, from 0 (identical to the original) to 10 (very different), to the original model and an MSDM distance value ($\in [0, 1]$). A cumulative Gaussian psychometric curve is then used to provide a nonlinear mapping between the objective and subjective scores.

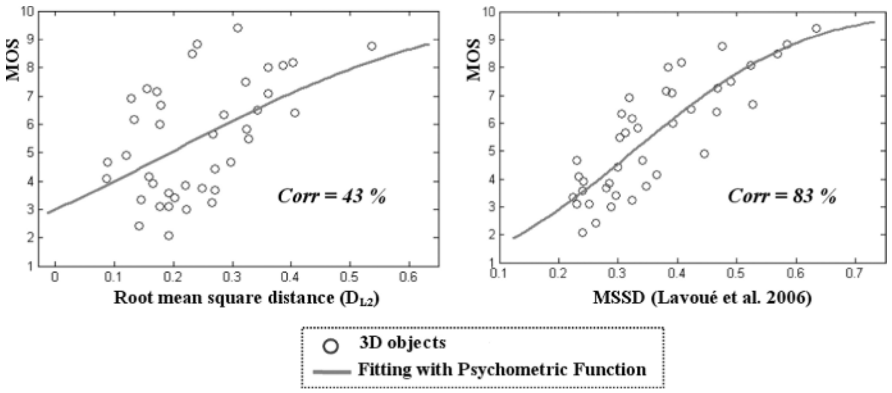


Figure 1.24 Subjective MOS vs. MSDM metric and the root mean square distance (D_{L2}) for 42 test objects

Figure 1.23 provides a first intuitive example: the *noised* object in (b) is associated with a higher MOS than the *smoothed* object in (d) (8.8 vs. 8.1). That seems intuitively normal since the smoothed model appears visually less distorted than the noised one. The MSDM reflects well this subjective opinion since, for (b), it is also higher than for (d) (0.64 vs. 0.58). On the contrary, the geometric mean distance (see Equation (1.21)) does not reflect this subjective opinion at all (0.16 vs. 0.25).

Figure 1.24 presents results of the psychometric curve fitting (cumulative Gaussian) between subjective and objective scores (root mean square distance (L_2) and MSDM). The correlation (Spearman) is better for MSDM (83 % vs. 43 %) which predicts well the subjective scores (MOS).

1.4 Summary

Technological advances in the fields of telecommunications, computer graphics and multimedia during the last decade have contributed to the evolution of digital data being manipulated, visualized and transmitted over the Internet. Nowadays, 3D data constitute the new emerging multimedia content.

However, 3D objects are more complex to handle than other multimedia data, such as audio signals or 2D images, since there are many different representations for such objects.

Three representation schemes are mainly used:

- **Polygonal mesh**: with this model, the object is represented by its boundary surface which is composed of a set of planar faces. Basically a polygonal mesh contains a set of 3D points (*vertices*) which are linked by *edges* to form a set of polygonal *facets*.
- **Parametric surface**: this family of 3D surfaces (including Bézier, B-spline and NURBS surfaces) is used particularly for computer-aided design (CAD). This model enables us to define mathematically exact surfaces, contrary to polygonal meshes which only represent an approximation.
- **Discrete representation**: this model does not represent the 3D object in Euclidean space, but in a 3D grid similar to the 2D image representation. Each element of the grid is a voxel (for volumetric pixel). Thus the object is represented by the set of voxels constituting its volume.

Several other representation models exist (implicit surface, subdivision surface, superquadric, CSG, etc.), but their utilization concerns more specific domains.

This 3D content comes from two main sources:

- The *acquisition*, using a 3D scanner or tomography for instance, which reproduces a real-world object, like a camera does for a 2D image.
- The *manual creation*, using specific software, similar to a painting of a 2D image.

The 3D models are subject to a wide variety of processing operations such as compression, simplification, watermarking or indexing, which may introduce some geometric artifacts on the shape. The main issue is to maximize the compression/simplification ratio or the watermark strength while minimizing these visual degradations.

In this context the need for efficient tools to measure the loss of quality or the visual difference between 3D objects becomes critical. However, classical metrics based on geometric differences like the Hausdorff distance do not match well with human visual perception. This perceptual gap has just started to be investigated for 3D models.

References

Akkouche S and Galin E 2001 Adaptive implicit surface polygonization using marching triangles. *Computer Graphics Forum* **20**(2), 67–80.

Alexa M, Gross M, Pauly M, Pfister H, Stamminger M and Zwicker M 2004 Point-based computer graphics. *ACM SIGGRAPH course notes*, p. 7.

Angelidis A and Cani MP 2002 Adaptive implicit modeling using subdivision curves and surfaces as skeletons. *Solid Modelling and Applications*, pp. 45–52.

Aspert N, Santa-Cruz D and Ebrahimi T 2002 Mesh: measuring error between surfaces using the Hausdorff distance. *IEEE International Conference on Multimedia and Expo (ICME)*, pp. 705–708.

Barbier A, Galin E and Akkouche S 2005 A framework for modeling, animating, and morphing textured implicit models. *Journal of Graphical Models* **67**(3), 166–188.

Bardinet E, Cohen LD and Ayache N 1994 Fitting 3D data using superquadrics and free-form deformations. *International Conference on Pattern Recognition*, pp. 79–83.

Barr AH 1981 Superquadrics and angle preserving transformations. *IEEE Computer Graphics and Applications* **1**(1), 11–23.

Barr AH 1984 Global and local deformations of solid primitives. *Computer Graphics* **18**(3), 21–30.

Bézier P 1972 *Numerical control: Mathematics and applications.* John Wiley & Sons' Ltd.

Biederman I 1985 Human image understanding: recent research and a theory. *Computer Vision, Graphics, and Image Processing* **32**, 29–73.

Binford T 1971 Visual perception by computer. *IEEE Conference on Systems and Control.*

Blanc C and Schlick C 1996 Ratioquadrics: an alternative model for superquadrics. *The Visual Computer* **12**(8), 420–428.

Blinn JF 1982 A generalization of algebraic surface drawing. *ACM Transactions on Graphics* pp. 235–256.

Bloomenthal J and Shoemake K 1991 Convolution surfaces. *Computer Graphics* **25**(4), 251–256.

Bloomenthal J and Wyvill B 1990 Interactive techniques for implicit modeling. *Computer Graphics* **24**(2), 109–116.

Bolin M and Meyer G 1998 A perceptually based adaptive sampling algorithm. *ACM SIGGRAPH*, pp. 299–309.

Catmull E and Clark J 1978 Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design* **10**(6), 350–355.

Cignoni P, Occhini C and Scorpigno R 1998 Metro: measuring error on simplified surfaces. *Computer Graphics Forum* **17**(2), 167–174.

Coeurjolly D and Klette R 2004 A comparative evaluation of length estimators of digital curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **26**(2), 252–258.

Cohen I and Cohen LD 1996 A hybrid hyperquadric model for 2D and 3D data fitting. *Computer Vision and Image Understanding* **63**(3), 527–541.

Corsini M, Drelie Gelasca E and Ebrahimi T 2005 A multi-scale roughness metric for 3D watermarking quality assessment. *Workshop on Image Analysis for Multimedia Interactive Services*.

Daly S 1993 The visible differences predictor: an algorithm for the assessment of image fidelity. In *Digital Images and Human Vision*, A. B. Watson, Ed. MIT Press, Cambridge, MA, 179–206.

DeCarlo D and Metaxas D 1998 Shape evolution with structural and topological changes using blending. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **20**, 1186–1205.

Doo D and Sabin M 1978 Behavior of recursive division surfaces near extraordinary points. *Computer Aided Design* **10**, 356–360.

Drelie Gelasca E, Corsini M and Ebrahimi T 2005 Objective evaluation of the perceptual quality of 3D watermarking. *IEEE International Conference on Image Processing*, pp. 241–244.

Dumont R, Pellacini F and Ferwerda J 2003 Perceptually-driven decision theory for interactive realistic rendering. *ACM Transactions on Graphics* **22**(2), 152–181.

Eckert M and Bradley A 1998 Perceptual quality metrics applied to still image compression. *Signal Processing* **70**(3), 177–200.

Farin G 1993 *Curves and surfaces for CAGD: A practical guide.* Academic Press.

Farrugia J and Peroche B 2004 A progressive rendering algorithm using an adaptive perceptually based image metric. *Computer Graphics Forum* **23**(3), 605–614.

Ferwerda J, Pattanaik S, Shirley P and Greenberg D 1997 A model of visual masking for computer graphics. *ACM SIGGRAPH*, pp. 143–152.

Flin F, Brzoska JB, Lesaffre B, Coléou C, Lamboley P, Coeurjolly D, Teytaud O, Vignoles G and Delesse JF 2005 An adaptive filtering method to evaluate normal vectors and surface areas of 3D objects. Application to snow images from X-ray tomography. *IEEE Transactions on Image Processing* **14**(5), 585–596.

Fougerolle Y, Gribok A, Foufou S, Truchetet F and Abidi M 2005 Boolean operations with implicit and parametric representation of primitives using R-functions. *IEEE Transactions on Visualization and Computer Graphics* **11**(5), 529–539.

Fougerolle Y, Gribok A, Foufou S, Truchetet F and Abidi M 2006 Radial supershapes for solid modeling. *Journal of Computer Science and Technology* **21**(2), 238–243.

Gielis J, Beirinckx B and Bastiaens E 2003 Superquadrics with rational and irrational symmetry. *ACM Symposium on Solid Modeling and Applications*, pp. 262–265.

Han S, Goldof D and Bowyer K 1993 Using hyperquadrics for shape recovery from range data. *IEEE International Conference on Computer Vision*, pp. 492–496.

Hoppe H, DeRose T, Duchamp T, Halstead M, Jin H, McDonald J, Schweitzer JE and Stuetzle W 1994 Piecewise smooth surface reconstruction. *ACM SIGGRAPH*, pp. 295–302.

Hornus S, Angelidis A and Cani MP 2003 Implicit modelling using subdivision-curves. *The Visual Computer* **19**(2–3), 94–104.

Howlett S, Hamill J and O'Sullivan C 2005 Predicting and evaluating saliency for simplified polygonal models. *ACM Transactions on Applied Perception* **2**(3), 286–308.

Jaklic A and Solina F 2003 Moments of superellipsoids and their application to range image registration. *IEEE Transactions on Systems, Man, and Cybernetics* **33**(4), 648–657.

Jaklic A, Leonardis A and Solina F 2000 *Segmentation and Recovery of Superquadrics*. Kluwer Academic.

Karni Z and Gotsman C 2000 Spectral compression of mesh geometry. *ACM SIGGRAPH*, pp. 279–286.

Keren D, Cooper DB and Subrahmonia J 1994 Describing complicated objects by implicit polynomials. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **16**(1), 38–53.

Kim S, Kim S and Kim C 2002 Discrete differential error metric for surface simplification. *Pacific Graphics*, pp. 276–283.

Kobbelt L 1996 Interpolatory subdivision on open quadrilateral nets with arbitrary topology. *Computer Graphics Forum* **15**(3), 409–420.

Kobbelt L and Botsch M 2004 A survey of point-based techniques in computer graphics. *Computers and Graphics* **28**(6), 801–814.

Kobbelt LP, Botsch M, Schwanecke U and Seidel HP 2001 Feature-sensitive surface extraction from volume data. *ACM SIGGRAPH*, pp. 27–66.

Krishnan S and Manocha D 1997 An efficient surface intersection algorithm based on lower-dimensional formulation. *ACM Transactions on Graphics* **16**(1), 74–106.

Lavoué G, Drelie Gelasca E, Dupont F, Baskurt A and Ebrahimi T 2006 Perceptually driven 3D distance metrics with application to watermarking. *SPIE Applications of Digital Image Processing XXIX*.

Lee C, Varshney A and Jacobs D 2005 Mesh saliency. *ACM SIGGRAPH*, pp. 659–666.

Loop C 1987 Smooth subdivision surfaces based on triangles. Master's thesis, Utah University.

Lorensen WE and Cline HE 1987 Marching cubes: a high resolution 3D surface construction algorithm. *Computer Graphics* **21**(4), 163–168.

Luebke D and Erikson C 1997 View-dependent simplification of arbitrary polygonal environments. *ACM SIGGRAPH*, pp. 199–208.

Luebke D and Hallen B 2001 Perceptually driven simplification for interactive rendering. *Eurographics Workshop on Rendering Techniques*, pp. 223–234.

Marc Levoy, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, Jeremy Ginsberg, Jonathan Shade, and Duane Fulk. The Digital Michelangelo Project: 3D Scanning of Large Statues. Proceedings of ACM SIGGRAPH 2000. pp. 131–144,

Marziliano P, Dufaux F, Winkler S and Ebrahimi T 2004 Perceptual blur and ringing metrics: Application to JPEG2000. *Signal Processing: Image Communication* **19**(2), 163–172.

Montiel E, Aguado AS and Zaluska E 1998 Surface subdivision for generating superquadrics. *The Visual Computer* **14**(1), 1–17.

MPEG-4 2002 ISO-IEC 14496-16. Coding of audio-visual objects: animation framework extension (AFX).

Murakami S and Ichihara H 1986 On a 3D display method by metaball technique. *Transactions of the Institute of Electronics, Information and Communication Engineers of Japan* **J70**(8), 1607–1615.

Muraki S 1991 Volumetric shape description of range data using *Blobby Model*. *Computer Graphics* **25**(4), 227–235.

Nguyen QL and Levine MD 1996 Representing 3D objects in range images using geons. *Computer Vision and Image Understanding* **63**(1), 158–198.

Nishimura H, Hirai M, Kawai T, Kawata T, Shirakawa I and Omura K 1985 Object modeling by distribution function and a method of image generation. *Transactions of the Institute of Electronics and Communication Engineers of Japan* **J68**(4), 718–725.

Pan Y, Cheng I and Basu A 2005 Quality metric for approximating subjective evaluation of 3-D objects. *IEEE Transactions on Multimedia* **7**(2), 269–279.

Peters J and Reif U 1997 The simplest subdivision scheme for smoothing polyhedra. *ACM Transactions on Graphics* **16**(4), 420–431.

Ponce J, Chelberg DM and Mann WB 1989 Invariant properties of straight homogeneous generalized cylinders and their contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **11**(9), 951–966.

Reddy M 1996 Scrooge: Perceptually-driven polygon reduction. *Computer Graphics Forum* **15**(4), 191–203.

Reddy M 2001 Perceptually optimized 3D graphics. *IEEE Computer Graphics and Applications* **21**(5), 68–75.

Rondao-Alface P, De Craene M and Macq B 2005 Three-dimensional image quality measurement for the benchmarking of 3D watermarking schemes. *Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents*, pp. 230–240.

Salvi J, Matabosch C, Fofi D and Forest J 2007 A review of recent range image registration methods with accuracy evaluation. *Image and Vision Computing*, **25**(5), 578–596.

Schaefer S and Warren J 2005 On C^2 triangle/quad subdivision. *ACM Transactions on Graphics* **24**(1), 28–36.

Sivignon I, Dupont F and Chassery JM 2004 Decomposition of a 3D discrete object surface into discrete plane pieces. *Algorithmica* **38**, 25–63.

Stam J and Loop C 2003 Quad/triangle subdivision. *Computer Graphics Forum* **22**(1), 79–85.

Terzopoulos D and Metaxas D 1991 Dynamic 3D models with local and global deformations: deformable superquadrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **13**(7), 703–714.

Tian D and AlRegib G 2004 FQM: a fast quality measure for efficient transmission of textured 3D models. *ACM Multimedia*, pp. 684–691.

Vaerman V, de Sola Fabregas C and Menegaz G 1997 The hyperquadrics: an efficient parametric surface representation. Technical Report LTS 97.10, Signal Processing Lab, Swiss Federal Institute of Technology.

Wang Z, Bovik A, Sheikh H and Simoncelli E 2004 Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing* **13**(4), 1–14.

Warren J and Weimer H 2002 *Subdivision methods for geometric design: A constructive approach.* Morgan Kaufmann.

Williams N, Luebke D, Cohen J, Kelley M and Schubert B 2003 Perceptually guided simplification of lit, textured meshes. *ACM Symposium on Interactive 3D Graphics*, pp. 113–121.

Wyvill B, Guy A and Galin E 1999 Extending the CSG tree (warping, blending and boolean operations in an implicit surface modeling system). *Computer Graphics Forum* **18**(2), 149–158.

Wyvill G, McPheeters C and Wyvill B 1986 Data structure for soft objects. *The Visual Computer* **2**, 227–234.

Zeroug M and Nevatia R 1996 Three-dimensional descriptions based on the analysis of the invariant and quasi-invariant properties of some curved-axis generalized cylinders. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **18**(3), 237–253.

Zhou L and Kambhamettu C 2001 Extending superquadrics with exponent functions: modeling and reconstruction. *Journal of Graphical Models* **63**(1), 1–20.

Zorin D, Schroder P and Sweldens W 1996 Interpolating subdivision for meshes with arbitrary topology. *ACM SIGGRAPH*, pp. 189–192.