Geometry Simplification

Carlos Andújar

Universitat Politècnica de Catalunya, Dept. LSI, Diagonal 647
E-08028 Barcelona, Spain
andujar@lsi.upc.es

February 4, 1999

Abstract

In this work we present the principles and applications of geometry simplification, focusing on simplification of polygonal representations of solids and surfaces. Related concepts such as multiresolution, level-of-detail and geometry compression are also discussed. A characterization of surface simplification methods is presented, including a classification, review and evaluation of the more relevant methods.

Contents

1 Introduction

Using a one-million polygon model to display an object that covers a few pixels on the screen is both unnecessary and inefficient. Even for close-ups of the object, our application might not require so much accuracy, or simply the model is too much complex to be rendered at interactive rates. Similar examples can be found not only in visualization but also in interference detection, visibility analysis, transmission of models over networks, acoustic modeling, geometric querying and reverse engineering. All these tasks have in common that they can be processed more efficiently using simplified representations of the objects whenever exact accuracy is not required.

Geometry simplification deals with generation of geometric models that resemble the input model but involve less faces, edges and vertices. *Level of Detail* is concerned to the possibility of using different representations of a geometric object having different levels of accuracy and complexity. *Multi-resolution models* provide several level-of-detail representations of a geometric model and have become a powerful tool in many computer graphics applications, including CAD, virtual reality and scientific visualization, as they can accelerate the handling of complex models by omitting unessential computation steps and reducing storage space.

Although multiresolution models are often obtained interactively [HG94], extensive research is being performed in developing algorithms for the automatic generation of LOD representations.

This work presents the principles and applications of geometry simplification, including topics such as multiresolution models, level of detail and automatic simplification methods for polyhedral models and triangle meshes.

The next section contains the geometric modeling background necessary for the rest of sections. Section 3 defines the scope of geometry simplification and related concepts such as geometry compression, multiresolution and representation scheme conversion. Section 4 deals with general and specific applications of geometry simplification, compression and multiresolution. Model requirements for visualization, progressive transmission and geometric operations are also discussed. The principles of geometry simplification are presented in Section 5, including error control, reduction measurement and topology preservation issues. Section 6 introduces a new characterization of automatic geometry simplification methods. Surface simplification strategies are discussed in Section 7. A classification of surface simplification methods is proposed in Section 8, where fundamental simplification algorithms are reviewed and characterized according to the proposed criteria. Finally, a comparison and evaluation of relevant algorithms is presented in Section 9, which also discusses current limitations and new trends of current surface simplification algorithms.

2 Geometric modeling

2.1 Aims and branches of geometric modeling

A *model* is an artificially constructed object that makes the observation of another object easier. Models are useful because certain characteristics of an object can be studied more easily regarding the model than its physical counterpart. *Geometric modeling* deals with representation and processing of geometric information on n-dimensional objects [Man88]. From the point of view of geometry simplification, the following branches of geometric modeling are specially relevant:

- *curve modeling*

  Curve modeling is devoted to the representation of *curves* one-dimensional curves, usually embedded in the plane or 3D space.

- *surface modeling*

  Surface modeling studies the representation of two-dimensional surfaces. Surface models give detailed information on the geometry of a curved surface, but do not always give sufficient information for determining all geometric properties of the object potentially bounded by the surface.

- *solid modeling*

  Solid modeling deals with *complete* and *valid* geometric representations of 3D objects whose interior is considered to be homogeneous and isotropic. By complete we mean that representations must be adequate for answering arbitrary geometric questions about the object [Man88]. By valid we mean bounded, regular sets with a two-manifold boundary [Man88].

- *volume modeling*

  Volume modeling deals with representation of spatial properties of heterogeneous, anisotropic 3D objects. Volume models are broadly used in medicine, earth sciences, biochemistry, biology, and fluid dynamics.

- *3D modeling*

  3D modeling is devoted to representation of geometric and non-geometric information on 3D objects. A *3D model* is an object model [Man88] including both geometric information (the geometric model) and nongeometric information, such as visual information.

2.2 Three-level view of modeling

A rigorous view of modeling is based on distinguishing between three separate levels of modeling:

- *physical objects*

  The aim of modeling is to study and argue about some real or imaginary things of our world.

- *mathematical models*

  A *mathematical model* is a geometric model that has a clear and intuitive connection with its physical counterpart. Mathematical models are suitable for human reasoning but often inappropriate for computer manipulation.

- *representations*

  A *representation scheme* is a set of rules defining the mapping from a mathematical model to another model suitable to computer manipulation. Such geometric model is called *representation*, and consists of a finite collection of basic elements called a *symbols*. The *domain* is the set of mathematical models that can be represented with a representation scheme. The extension of the domain depends on the expressive power of the representation scheme.

2.3 Mathematical models of solids and surfaces

A *two-manifold with boundary* is a topological space where every point has a neighborhood topologically equivalent to an open disk of the two-dimensional euclidean space E^2, except points on the edge of an open surface patch. Intuitively speaking, two-manifolds are non-selfintersecting, open surfaces. Two-manifolds with boundary are the most common mathematical model for surfaces not enclosing a volume.

A *two-manifold* is a topological space where every point has a neighborhood topologically equivalent to an open disk of E^2. Intuitively, two-manifolds are non-selfintersecting, closed surfaces. A two-manifold is said to be *realizable* if it encloses a 3D volume [Man88]. Orientable two-manifolds are the most common mathematical models for surfaces bounding a volume, and hence for solids.

2.4 Representation of solids and surfaces

Several representation schemes have been proposed in the literature for representing solids and surfaces.

Constructive models represent a point set as a combination of primitive point sets. Each primitive is represented as an instance of a primitive type. Combination operations are set boolean operations (union, intersection, difference, complement).

Boundary models [Man88] represent point sets in terms of its boundary. Objects are represented by dividing their surface into a collection of connected components called *faces*. The division is performed so that each face has a compact mathematical representation, e.g. the face lies on a planar, quadratic or parametric surface. The portion of the underlying surface that forms the face is trimmed out in terms of closed curves lying on such surface. This kind of representation is called *boundary representation* (BRep for short).

Space decomposition models (SDM for short) represent a point set as the union of disjoint regions of the space called *cells*. Cells containing part of the object are labeled as *black* and the rest are labeled as *white*. Unlike BRep models, SDM are suitable for representing volume data. On this case, cells represent regions with homegeneous interior with respect to the studied property. SDM are approximate models, as they cannot represent exactly most solids and surfaces (see [Req80], [Sam90b], [Sam90c]); the accuracy of the SDM depends on the size of the cells. Tetrahedra, cubes and boxes are the more relevant cell geometries. SDM play a special role in geometry simplification because solids and surfaces represented by these models can be trivially simplified, just gluing adjacent cells. Furthermore, SDM provide a simple and stable way of changing surface topology.

2.5 Representation of visual information

The most common surface appearance properties of 3D models intended for real-time visualization are:

- *color (diffuse, ambient and specular reflection coefficients)*

  Color is often defined in a per-object or per-face basis (final color depends on lighting calculations) or per-vertex (radiosity output).

- *texture images*

  Textures are often defined per-object or per-face.

- *texture coordinates*

  Explicit texture coordinates are defined at corners or vertices. Implicit texture coordinates (i.e. texture mapping functions) are defined at faces or objects.

- *surface normals*

  Real-time visualization systems use per-vertex normals for lighting computations.

2.6 Polyhedral representations

A *polyhedral model* is a boundary model whose faces are connected subsets of the plane. Each planar face is represented by one or more planar polygons. One polygon defines the outer boundary of the face, while the others (called *rings* or *interior loops*) represent interior holes of the faces. Polygons are described by an ordered sequence of straight line segments called *edges*. Each edge is defined by its two endpoints, called *vertices*.

A *polygonal mesh* is a polyhedral model whose faces are simply-connected, i.e. they can be represented by a single polygon.

A *triangle mesh* is a polygonal mesh whose faces are triangles (see Figure 2).

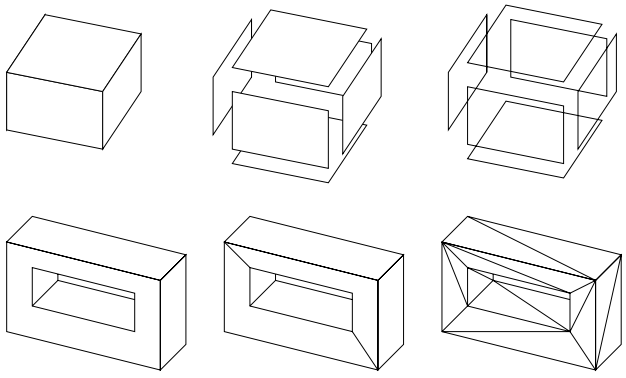


Figure 1: *Top*: Geometric entities of a BRep; *Bottom*: a polyhedral model (*left*), a polygonal mesh (*middle*) and a triangle mesh (*right*)

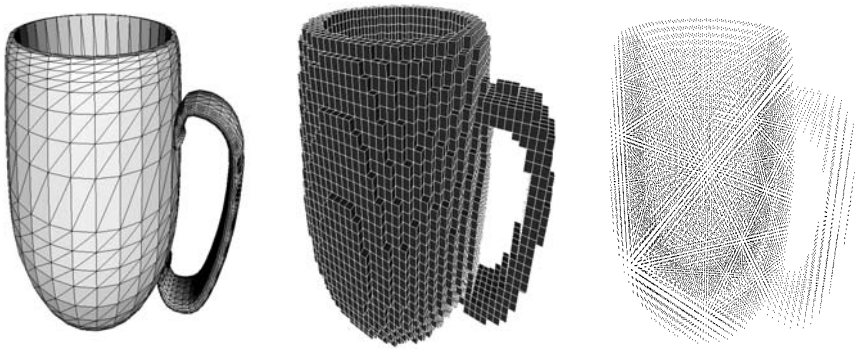


Figure 2: Rendering of a mug represented with a triangle mesh (*left*), voxel decomposition (*midlle*) and digital picture (*right*)

2.6.1 Voxel-based representations

A *voxel decomposition* is a SDM whose cells are equal-sized cubes, called *voxels*, arranged in a regular array. Voxel decompositions are suitable for representing solid objects and volume data. Solid objects are represented by labeling interior voxels as *black* (or 1) and voxels outside the solid as *white* (or 0) (see Figure 2). For volume data representation, voxels are labeled according to the the value of the property being modeled inside the voxel. In voxel decompositions, the interior of voxels is considered to be homogeneous.

A *3D picture* is a set of points arranged in a regular grid defining equal-sized cubic cells. Points of a 3D picture (called *lattice points*) are labeled according to the property being studied. The cubic cell defined by eight neighbor points is also called *voxel*. Unlike voxel decompositions, 3D pictures deal with voxels with non-homogeneous interior, since the property is only known at the lattice points, which coincide with voxel's vertices (see Figure 2). The space of interest can be conveniently scaled so that lattice points have integer coordinates. A *3D digital picture* is a set $B \subset Z^3$. The elements of Z^3 are called points of the picture. The points in B are called the *black points* of the picture; the points in $Z^3 - B$ are called the *white points* of the picture.

Two points in 3D-space are said to be *26-adjacent* if they are distinct and each coordinate of one differs from the corresponding coordinate of the other by at most 1; two points *18-adjacent* if they are 26-adjacent and differ in at most two of their coordinates; two points *6-adjacent* if they are 26-adjacent and differ in at most one coordinate. In terms of voxels, 26-adjacent voxels share a face, edge or vertex; 18-adjacent voxels share a face or edge, and 6-adjacent voxels share only face. An n-neighbor of a point (resp. voxel) p is a point (resp. voxel) that is n-adjacent to p (see Figure 3). A set S of points is *n-connected* if S cannot be partitioned into two sets that are not n-adjacent to each other.

2.6.2 Octree-based representations

The *octree* representation uses a recursive subdivision of a cubic universe into eight octants that are arranged into an 8-ary tree. In the classical octree representation [Sam90a] (CO for short), each node consists of a code (often called *color*) and eight pointers towards eight sons. Nodes corresponding to cubic regions completely inside the object are labeled as *black* (B). Nodes corresponding to cubic regions completely outside the object are labeled as *white* (W). White and black nodes are leaves, i.e. they are no further subdivided. Nodes

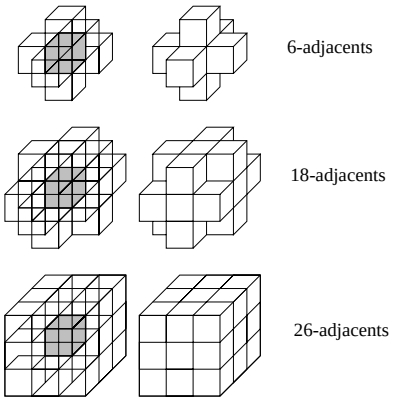


Figure 3: A voxel (*shaded*) and its 6,18,26-neighbors

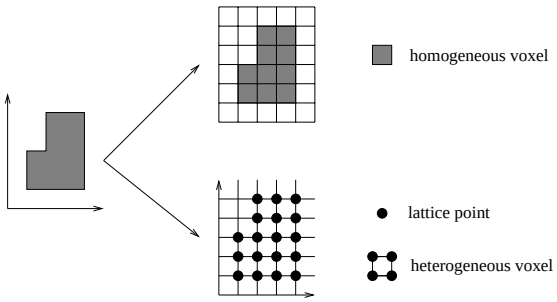


Figure 4: Voxel decompositions and 3D pictures

containing a part of the boundary are labeled as *grey* (G) and are recursively subdivided. Leaf grey nodes are called *terminal grey* (TG) nodes.

The *maximal division classical octree* [BJN+88] (MDCO for short) is an extension of the classical octree scheme where all terminal grey nodes belong to the same level of the octree, and hence have the same size. Given a solid P and a non-negative integer l, $MDCO(P, l)$ is an octree representation of P, containing W, B, G and TG nodes, where all TG nodes belong to the deepest level l.

The boundary of the solid is completely contained in the set of TG nodes. The set of TG nodes can be viewed as a voxelization of the object's boundary, with the hierarchical structure of the MDCO on top of it. Unlike voxel-based representations, the adaptive decomposition provided by octrees cells allows compression of data in homogeneous regions.

2.6.3 Surface reconstruction

Isosurface extraction deals with generation of isosurfaces from volume data. Isosurfaces approximate the points with a given property value, called *isodensity value*. Isosurface extraction is a powerful tool for analysis and visualization of volume data. *Surface fitting* deals with generation of surfaces approximating a set of points which are known to be on the surface. These points are usually acquired from 3D digitizing techniques. Isosurface extraction, surface fitting and SDM to BRep conversion are globally known as *surface reconstruction*, i.e. surface generation from some kind of spatial data.

3 Scope of geometry simplification

3.1 Definition of geometry simplification

Geometry Simplification studies the simplification of geometric models, i.e. the generation of geometric models that resemble the input model but involve less symbols. In order to be called simplification, the representation scheme must be preserved.

3.2 Components of a simplification process

The geometric model to be simplified is called *input model*, and the underlying physical object represented by this model is called *original object*. The result of the simplification process is referred as *output* or *simplified* model. The input and the output models are slightly different *representations* of the original object differing in complexity and accuracy.

The *approximation error* is a quantification of the difference between the input model and the simplified model. The metric used for such a quantification is called *error metric*. When the error metric is expensive to compute, cheap *error estimations* are used instead.

Since geometric fidelity and symbol reduction are opposite goals, the amount of reduction of the simplification process is defined by user-defined parameters called *reduction parameters*.

3.3 Branches of geometry simplification

Depending on the dimension of the original object, several kinds of geometry simplification must be distinguished:

- *curve simplification*

  Curve simplification deals with simplification of curves embedded in a two-dimensional or a three-dimensional space.

- *surface simplification*

  Surface simplification deals with simplification of surfaces embedded in a three-dimensional space.

- *solid simplification*

  Surface simplification deals with simplification of three-dimensional solids.

- *volume simplification*

  Volume simplification deals with simplification of volume models.

Two branches of surface simplification are specially relevant:

- *terrain simplification*

  Terrain simplification deals with simplification of digital terrain models (DTM).

- *isosurface simplification*

  Isosurface simplification deals with simplification of boundary models generated by isosurface extraction techniques from volume data.

Furthermore, surface simplification can be classified according to the representation scheme used for the input and output models:

- *arbitrary polyhedra simplification*

  Polyhedra simplification deals with simplification of polyhedral models of solids and surfaces.

- *triangle mesh simplification*

  Triangle mesh simplification (mesh simplification for short) deals with simplification of triangle mesh representations of solids and surfaces.

3.4 Related disciplines

3.4.1 BRep construction from other schemes

As we pointed out in definition of geometry simplification (Section 3.1), the representation scheme of the input model is preserved through the simplification process. For instance, if the input model is a polygonal mesh, the output model must be a polygonal mesh too. When the representation schemes of the input and output models are different, then we call the process *scheme conversion* instead of simplification. Scheme conversion is closely related to simplification because sometimes is an alternative for generating level-of-detail representations of an object. *Evaluation* is a kind of scheme conversion specially important for level-of-detail generation of curve, surface and solid models. Evaluation is the conversion from an implicit model into an explicit model. Examples of implicit models are parametric representation of curves (Bezier, Splines...), surfaces (NURBS...) and solids (CSG). Generation of level-of-detail boundary representations from an implicit model is generally straightforward since these models support the evaluation with different levels of accuracy. For instance, the spacing between samples in parameter space defines the accuracy of the evaluation of a parametric curve or surface. Similarly, level-of-detail representations of primitive solids such as spheres, cones and cylinders are straightforward to generate.

If both a parametric and an explicit representation are available as input, the evaluation usually gives better results when trying to generate accurate approximations of simple objects; for generating coarse representations of complex objects, topology simplification techniques are required.

3.4.2 Geometry compression

A concept quite related to geometry simplification is geometry compression. Geometry simplification preserves the representation scheme, so symbol reduction (vertices, edges, loops, faces) is achieved at the expense of accuracy, and hence simplification is generally lossy.

Unlike geometry simplification, *geometry compression* deals with loss-less coding of geometric models. A common example of geometry compression of polygonal meshes consists of storing vertex coordinates in a separate array, and represent polygons as ordered sequences of indices to the vertex array instead of the actual coordinates. A still more efficient coding of a triangle mesh is the triangle strips coding [HB94].

Some compressed schemes, such as triangle strips, have a decompression so efficient that can be decompressed in real-time (even by specialized hardware). Other compression schemes are used only for efficient storing and transmission of models, since they require an expensive

decompression or appropriate API's are not available. Unlike geometry simplification, the number of symbols of the input model is preserved through the compression/decompression cycle.

3.4.3 Other kinds of simplification in computer graphics

Yet it is out of the scope of this work, it is worth to mention other kinds of compression and simplification used in computer graphics.

- *still image simplification and compression*

  Run-length encoding is a simple and common loss-less compression scheme for still images. An example of lossy compression was defined by JPEG [Joi90]. See [Cla95] for a review of compression of still images.

  Unlike image compression, simplification of images is achieved at the expense of image resolution and involves minimizing filters.

- *video compression*

  Extensive work has been done in lossy and loss-less compression of digital video. See [Cla95] for a review of video compression schemes such as MPEG [MPFL96], [YL95].

3.5 Level of detail and multiresolution models

Level of Detail is concerned to the possibility of using different representations of a geometric object having different levels of accuracy and complexity.

Multiresolution models provide several level-of-detail representations of a geometric model. Such models have been proposed for efficient handling of many geometric entities: curves (specially in GIS), surfaces (mainly in computer graphics, virtual reality and GIS), solids (computer graphics, CAD, virtual reality and object recognition) and volume data (scientific visualization).

Multiresolution models must support the query for extracting a specific representation. Several architectures have been proposed in the literature:

- *collection of LOD's*

  A *collection multiresolution model* maintains a simple collection of independent LOD representations. The extraction of a specific representation is straightforward. Since each representation is stored independently, the memory requirements is about the sum of the sizes of individual representations.

- *hierarchical*

  A *hierarchical multiresolution model* maintains relations between consecutive LOD representations, so extraction of a specific representation requires several levels to be accessed.

- *incremental*

  An *incremental multiresolution model* maintains a compact structure from which different LOD representations can be extracted. Usually this is achieved by storing a coarse representation and an ordered sequence of local updates. Extraction proceeds through the iterative application of updates starting from the coarse representation, until the representation has the required accuracy.

Regardless of the multiresolution model, the scene can adopt different structures:

- per-scene LOD

  The scene model involves a collection of LOD representations, each one containing a description of all the objects in the scene. Coarse representations are used for interactive tasks and accurate representations are used when accurate results are required.

- *per-object LOD*

  The scene is represented by a tree or DAG. Terminal nodes correspond to individual objects which are described by several LOD representations. The main advantage over the per-scene approach is that each object can be processed at the optimum resolution depending on object-specific context.

- *per-group LOD*

  Like in the per-object structure, the scene is represented by a tree or a DAG, but LOD representations are used not only for individual objects but also for groups. Existence of LOD representation at intermediate levels provides a convenient way of representing coarse representations of complex objects. For example, the coarser representation of a house can be a textured box.

Generation of multiresolution models usually requires using a simplification algorithm for the generation of the LOD representations with varying reduction parameters.

3.6 LOD-based algorithms

LOD-based algoriths use multiresolution models to process geometric entities at the optimum resolution depending on application-specific contexts. For example, in real-time visualization, small, distant objects are displayed using coarse representations while close objects are displayed at full accuracy. In interactive collision detection, simplified representations are used as a balanced solution between bounding boxes and the accurate representation.

The components of a LOD-based algorithm are a multiresolution model, and a selection heuristic. The *selection heuristic* provides a mechanism to select the LOD representation to be extracted from the multiresolution model according to specific application and context requirements.

An algorithmic paradigm related to LOD-based processing is that of *context-varying geometric processing*. Such algorithms provide several ways to process geometric data depending on accuracy and speed requirements. For instance, advanced display effects such as fog, anti-aliasing, texturing, and Gouraud shading are used or not depending on frame load. A more sophisticated example found in high-end graphic systems is dynamic resolution. Dynamic resolution is a dynamic adjustment of the virtual frame-buffer resolution (used in super-sampling-based anti-aliasing) in fill-limited applications in order to get a fixed frame rate. In these cases, the selection algorithm must select both the LOD representation and the processing method.

4 Applications

4.1 Operations over geometric models

As noted in definition of geometry simplification (Section 3.1), simplified representations involve less symbols (vertices, edges, loops and faces) than their accurate counterparts, so simplified versions can be stored, processed and transmitted more efficiently.

The following list contains the more relevant operations applied to geometric models. Many of these operations can be done more efficiently using simplified representations.

- *representation and storage*

  Geometric models need to be represented internally in main memory and stored in non-volatile media.

- *transmission*

  Transmission of geometric models through computer networks is becoming more common as the Internet and network-oriented standard formats such as VRML [ANM97] are being used abroad.

  Another kind of transmission, critical in high-end graphics hardware, is the communication through the bus connecting the host with the graphics pipeline.

- *real-time visualization*

  Visualization is used in conjunction with image-acceleration techniques: backface culling, viewing frustum culling, oclussion culling and LOD-based visualization. Oclussion culling requires visibility analysis preprocessing, and LOD-based visualization requires creation of multiresolution models.

- *photo-realistic rendering*

  Photo-realistic rendering usually requires sophisticated techniques which currently cannot be applied in real-time, such as global illumination, ray-tracing, particle dynamics and bump maps.

- *collision detection*

  Collision detection (also known as interference detection) is usually used in real-time visualization systems mock-up systems.

- *spatial sound generation*

  Spatial sound generation is important in many virtual reality applications. Computation of reverberation paths requires performing intersection tests between rays and relevant objects of the scene, such as walls and floors.

- *boolean operations*

  Boolean operations (union, intersection, difference, etc.) are a powerful tool for creating complex models from simple primitives.

- *selection*

  Picking is used in CAD and other applications for selecting the objects to be affected by future operations.

- *geometric queries*

  Geometric queries range from volume and area computation, through integrity tests up to point inside solid tests and ray-solid intersection.

- *scheme conversion*

  Geometric models often need to be converted to and from other representation schemes.

4.2 Pros and cons of using simplified representations

The potential gains of using simplified representations are:

- store models more efficiently, using less disk space to store them and less core memory to represent them internally;
- reduce transmission times while sending models over computer networks;
- enable handling of complex scenes in limited hardware;
- accelerate real-time visualization by reducing the amount of data processed in most stages of the graphics pipeline: scene traversal, culling, data delivery, transformation and rasterization. Scene traversal cost depends on the number of symbols of scene objects, so reducing this number improves performance. The same argument holds for backface culling (depending on the number of faces) and real-time illumination (depending on the number of vertices). Viewing frustum culling and oclussion culling are already based on simple bounding volumes and constant visibility cells, respectively. However, preprocess required for oclussion culling (visibility analysis) can be done more efficiently with simplified representations;
- accelerate photo-realistic rendering by reducing the amount of data being processed in scene traversal, vertex transformation, illumination computation, ray-tracing, etc.
- speed-up collision tests which depend on the number of symbols;
- faster geometric query processing, such as point inside solid test;
- accelerate spatial sound processing;
- apply boolean operations more efficiently. When the simplification process filters model degeneracies, boolean operations are also more robust;
- accelerate conversion to other representation schemes.

Simplification gains will not be eclipsed by future hardware developments because many geometric processing tasks have supra linear cost and because the size of geometric models is growing as long as more powerful hardware is becoming available. Advantages of simplified versions are more evident in processing tasks requiring supra linear running times since improvements in hardware performance have little impact in the amount of data that can be processed in real-time with such algorithms.

Unfortunately, there are several limitations of using simplified representations:

- *accuracy loss*

  Symbol reduction is achieved at the expense of accuracy loss; however, accuracy is frequently higher than is required in many contexts (e.g. visualization of small, distant objects).

- *non-geometric data loss*

  When simplifying 3D models, some non-geometric data may be lost during simplification, such as texture coordinates, per-vertex normals and surface properties.

4.3 Pros and cons of using compressed representations

The potential gains of using compressed representations are:

- store models more efficiently, using less disk space to store them and less core memory to represent them internally;
- dramatically reduce transmission times while sending models over computer networks;
- reduce bottlenecks while transmitting geometric data from the host to the graphics pipeline (e.g. triangle strips);
- accelerate per-vertex computations.

Like in simplification, there are several limitations of using compressed representations:

- compression and decompression stages introduce an overhead. When decompression cannot be done in real-time, compressed representations are used only for external storage and transmission over networks;
- some compression schemes are not fully loss-less, since they reduce vertex coordinates accuracy by vertex quantification or prediction. Although this accuracy loss is small compared to simplification techniques, integrity problems may appear when modifying vertex coordinates (e.g. two-manifold surfaces become non-manifold);
- many current compression schemes do not fully support non-geometric data.

4.4 Pros and cons of using multiresolution models

Multiresolution models are the basis for LOD-based processing. With a few exceptions discussed below, the gains of using simplified representations can be obtained using multiresolution models. The main difference is that accuracy of the representation can be adjusted in real-time according to application's requirements (e.g. need of interactive response) from the exact representation to a coarse one.

Per-object multiresolution models also allow this adjustment to be done in a per-object basis, according to application's context (e.g. camera position). Incremental multiresolution models allow progressive transmission of geometric models over networks.

Like the previous approaches, using multiresolution representations has some limitations:

- multiresolution models need more storage and memory space than their single representation counterparts, specially multiresolution models maintaining collection of independent representations, which require about the sum of sizes of the representations;
- only incremental multiresolution models can be transmitted efficiently;

- representation selection and extraction introduce an overhead. Selection is critical when its goal is to provide a fixed frame rate in real-time visualization. Extraction is more expensive in incremental multiresolution models;
- LOD-based processing requires multiresolution models to be generated. There are three ways of generating multiresolution models: interactively by hand [Cro82], [HG94] (i.e. modeling from scratch, as in old flight simulators), by automatic simplification and by scheme conversion. The multiresolution model is generally computed previously because existing methods are still too slow to be applied in real time.
- once created, multiresolution models are harder to update than single representations; for instance, color and textures changes must be spreaded over all LOD representations.

4.5 Specific contexts for geometry simplification

There are several contexts where simplification is specially important:

- *scenes inherently complex*

  Applications in areas such as automobile and shipbuilding design require huge models containing hundreds or millions of objects [AAB+97a].

- *verbose representations due to the acquisition process*

  We say that a representation is *redundant* when the represented object can be described with the same accuracy and in the same representation scheme using less symbols. An example of redundant representations are polyhedral models containing adjacent coplanar faces.

  A representation is said to be *verbose* when the represented object can be described with an application-acceptable accuracy in the same representation scheme using less symbols. An example of verbose representations are those containing adjacent, quasi-coplanar faces.

  Sometimes the complexity of the representation is due to the acquisition process. *Surface fitting* algorithms generate surface representations from spatial data acquired through laser scanners, magnetic-field digitizers, etc. *Isosurface extraction* algorithms generate surface representations from volume data acquired through CT-scan, MRI, etc. Usually these reconstruction methods produce verbose representations because they are non-adaptive, i.e. they produce representations with the same symbol density along the surface, regardless of surface curvature. Most reconstruction methods are unable to create large faces even in almost planar regions because face size is limited by grid spacing [Kal92].

- *limitations of the representation scheme*

  Polyhedral models have a limited expressive power; only polyhedral objects can be exactly represented. Objects involving curved faces are represented by tesselating the surface in planar patches. Objects with a mathematical description as much simple as a sphere require a large number of planar faces for accurate representation.

- *disparity between accuracy requirements*

  A geometric model is frequently used by many applications and users during the life cycle of the product. Objects are modeled with a high level of accuracy since these products are usually manufactured from CAD output. However, other tasks such as public presentations and walkthroughs do not require such accuracy. Another example

is the representation of a car required by an automobile design application compared to the that required by a traffic simulation application.

- *limitation of resources*

  Sometimes a geometric model cannot be processed by some applications because resource limitation: the model can be too large to fit in the memory of the workstation or communication lines might be too slow to provide remote access at reasonable rates.

- *need for interactive response*

  Many computer graphics applications require interactive output, but even high-end workstations are unable to interactively display the huge amounts of polygonal data available in industrial and academic areas. Frame rate is critical in applications such as virtual reality where head-mounted displays with low frame rates produce disorientation.

Simplification and multiresolution models have become a powerful tool in many areas, including CAD, GIS, virtual reality and scientific visualization, in tasks such as real-time visualization, progressive transmission, visibility analysis, acoustic modeling, geometric computation, verbosity reduction of reconstructed surfaces and multiresolution interactive modeling.

A few examples of specific applications include shipbuilding design, automobile design, vehicle simulators and VR walkthroughs.

4.6 LOD-based real-time visualization

The most obvious application of multiresolution models is found in real-time visualization due to the range from which an object can be seen in an interactive flythrough. Objects that cover a small portion of the screen can be displayed using coarse representations without seriously affecting image quality. Accurate representations are reserved for close, important objects. The need of multiresolution objects was originally stated in [Cla76] and discussed in [Cro82], where a representation selection based on the number of pixels covered by the bounding box of the object is proposed.

LOD-based real-time visualization is one of the image acceleration techniques. Another software-based image acceleration technique is oclusion culling, based on visibility analysis. Visibility analysis calculates those portions of the scene potentially visible from a specific point of view and store them in an appropriate structure which is accessed in real time for occlusion culling [FvDFH90]. Most visibility processing approaches are based on space subdivision in constant visibility cells [FST92], [TH93], [Smi94].

As any LOD-based processing, LOD-based real-time visualization requires a multiresolution model and a selection algorithm.

4.6.1 Representation selection

Representation selection in real-time visualization may pursue three different goals:

- *constant frame rate*

  the goal is to provide a constant frame rate trying to keep image quality as much as possible; the minimum frame rate is defined by the user;

- *constant image quality*

  the goal is to display the scene as fast as possible while maintaining a fixed, user-defined image quality;

- *optimum ratio*

  the goal is to provide an optimum ratio between image quality and frame rate; a user-defined parameter weights these opposite goals.

Multiresolution models must contain information for the selection algorithm such as the approximation error of each representation. In visualization-oriented multiresolution models, a list of ranges at which each LOD representation must be displayed can be provided instead of approximation errors. Ranges are defined by the minimum and maximum distances to the observer for a given representation. This approach is adopted in VRML [ANM97] and Open Inventor LOD-nodes.

Selection algorithms can be classified as follows [FS93]:

- *static selection*

  Static selection is based on estimating image error from the approximation error of each representation and the camera settings of the current frame. Static selection does not take into account system performance and graphics pipeline load. It is useful for providing constant image quality, but frame rate is not fixed neither reasonably bounded.

- *dynamic selection*

  Dynamic selection (also known as system-adaptive) takes into account current system performance, so frame rate homogeneity is increased. Dynamic selection can be reactive or predictive. Reactive selection uses history data about rendering times of previous frames to increase or decrease the threshold used for selecting representations as in the static approach. Performance of LOD-based rendering based on reactive selection is highly dependent on frame coherence. Predictive selection tries to estimate future image complexity [FS93] in order to provide a fixed frame rate.

Regardless of the selection algorithm, there are several heuristics to evaluate the impact of using a simplified representation in the image quality:

- *pixels covered*

  Pixels covered by the object on the screen is the most common and simple heuristic used for LOD selection. This area is estimated by projecting its bounding box over the screen plane.

- *application-specific importance*

  Some objects have special meaning for the application. For instance, background objects in a crane simulator are far less important than cables and containers.

- *proximity to focus area*

  This heuristic is based on distribution of photosensitive cells in the human retina. Assuming that human interest is centered on the center of the displayed image, objects in the periferia can be displayed using simplified representations. This heuristic is specially useful when image is displayed through immersive systems such as head-mounted displays and CAVE's.

- *speed of the object*

  Objects than move rapidly with respect to the observer are hardly perceived so greater image error is acceptable.

4.6.2 Model requirements

Multiresolution models to be used only for real-time visualization have specific properties:

- *model integrity relaxation*

  real-time visualization applications are somewhat less concerned with model integrity than other operations, so in many visualization applications, model integrity is not enforced: interpenetrating faces, repeated vertices, non-manifold boundaries and other model degeneracies are not relevant unless they impact the resulting image.

- *visual information*

  visualization in some applications require a high degree of realism, so material properties are combined with geometric models to produce 3D models. LOD-based visualization requires not only geometric approximation but also appearance preservation.

- *use of textures instead of complex shapes*

  An *impostor* is any 3D model that retains less visual information than the original but can be displayed more efficiently [MS95]. Impostors range from simplified representations through textured boxes up to billboards. *Billboards* are textured polygons which are dynamically oriented so they always face the observer [RH94]. Billboards are useful for representing complex shapes with axial symmetry.

Due to these specific properties, many simplification methods create visualization oriented models, such [RB93] which creates representations containing dangling points and lines.

4.6.3 The abrupt change problem

Image artifacts affect LOD-based visualization based on collection of LOD representations. Since each representation is stored separately, the number of levels of detail must necessarily be small, and hence changes between two consecutive levels are abrupt, causing undesirable effects during transition from a level to another. Human visual system is specially sensible to such popping effects.

There are many solutions to this problem. One approach consists of using simplified representations only when image impact is very low, so just a few pixels are involved. Unfortunately, this approach is incompatible with the goal of maintaining a fixed frame rate. Some high-end API's provide a mechanism to reduce these artifacts. In IRIS Performer [Eck97], an image-space smooth transition is achieved by a weighted blending of both representations in a small interval around the switch point.

Some surface simplification methods allow a smooth transition between two consecutive LOD representations [RB93], [Tur92], [HHK+95], [Eea95], [DLW94], using parameterized representations or incremental multiresolution models, but extraction overhead, which must be done in visualization time, may exceed the speed-up of the LOD-based visualization.

The previous techniques are not mutually exclusive, and both object-space and image-space smoothing techniques can be used together to reduce perception of image artifacts.

4.6.4 Appearance preservation

The final color of image pixels depends on many factors, some of them are 3D model-independent, such as the lighting model, properties of light sources, and atmospheric effects, and other are included in the 3D model. Visualization-oriented simplification is concerned with preservation of model-dependent visual properties:

- *normal preservation*

  The overall orientation of faces should be preserved, since on-screen color in most lighting models depends on surface normals. Radiosity models, which store final color information in a per-vertex basis, are an exception to these rule. In this case, the simplified representations must preserve the per-vertex or per-corner color information. A method for surface appearance simplification which do not require the preservation of overall orientation of faces is presented in [COM98]. The method is based on a parameterization of the triangulated surface and the storage of surface appearance attributes in normal and color maps. Although it achieves very realistic approximations of complex shapes and provides a bound in the resulting image error, normal maps are still available only in prototyped hardware [OL98], and texture maps increase rasterization cost and can decrease overall performance if the graphics subsystem is fill-limited.

- *material properties preservation*

  Other surface appearance attributes, such as color and texture, must be preserved. When these parameters come on a per-face basis, faces with different values for such attributes can be merged together only if the impact on the resulting image is acceptable. Unfortunately, surface simplification methods effectively handling such appearance attributes are still very rare (see [CPD+96], [Hop97]).

4.7 Simplification requirements for model transmission

Simplification, compression and multiresolution models have different applications in transmission of geometric models over computer networks, specially the Internet.

A simplified representation can be transmitted faster as it involves less symbols than the accurate representation, assuming that the simplified representation replaces the accurate one.

Compression techniques allow quick, loss-less transmission as they code geometry and incidence information using less bits than uncompressed representations.

Multiresolution models require more time to be completely transmitted, but they allow progressive transmission: a very coarse representation of a huge model can be downloaded in a few seconds. This very simple representation is refined as transmission proceeds until it becomes the accurate representation [NAM96].

Incremental multiresolution models are specially suitable for progressive transmission [Hop96] since total transmission time is not too much greater than the transmission time required by the accurate representation. Simplification algorithms based on multiresolution analysis (i.e. wavelets) [DLW94], [Eea95], [CPD+96] and progressive meshes [Hop96] are specially suitable for progressive transmission.

Multiresolution models storing a collection of independent LOD representations are also useful as they may contain a coarse representation used as a thumbnail of the model, from which the user decides whether download or not the accurate representation.

4.8 Simplification requirements for geometric queries

Simplified and multiresolution models have different applications in geometric operations such as collision detection, visibility analysis, etc.

A simplified representation can be processed faster as it involves less symbols than the accurate representation.

There are two ways of using simplified representations in geometric queries. In the lossy approach, all queries are processed using simplified representations, so reported results are always approximate. In the loss-less approach, average running times of query operations can be decremented by providing a quick answer report in simple cases. The precedents of this technique are bounding volumes such as bounding boxes and bounding spheres, which in fact are super-simplified representations. An example of loss-less query acceleration via simplified representation is found in collision detection. A preliminary test uses a simple bounding representation to quickly discard trivial cases of non-intersection.

In [Vel92] a hierarchical multiresolution model for geometric query acceleration is discussed. Collection of LODs based multiresolution models are being used for collision detection [AAB+97a].

4.9 Summary of application-dependent requirements

The following table shows the requirements of simplified, compressed and multiresolution models according to the target application:

Use	Model requirements
storage	compressed representations
host-graphics transmission	real-time decompression
transmission over networks	incremental recovering

Figure 5: Requirements for compression

Use	Model requirements
real-time visualization	appearance preservation
visibility analysis	bounded error
collision detection	bounded error
geometric queries	bounded error, integrity enforcement
reverberation sound	topology simplification

Figure 6: Requirements for simplified models

Use	Model requirements
real-time visualization	appearance preservation, smooth transition
visibility analysis	bounded error, bounding volumes
collision detection	bounded error, bounding volumes
progressive transmission	incremental multiresolution

Figure 7: Requirements for multiresolution models

5 Principles of geometry simplification

5.1 Error control and measure

As noted in definition of geometry simplification (Section 3.1), symbol reduction is achieved by simplification algorithms at the expense of accuracy.

The *approximation error* is a quantification of the difference between the input model and the simplified model. The metric used for such a quantification is called *error metric*. When the error metric is expensive to compute, cheap *error estimations* are used instead. Regardless of the error metric adopted, approximation error is measured in object coordinates and hence it is application-independent.

Processing geometric objects using simplified representations instead of accurate representations produces an error in the result, called *result error*, which is consequence of the approximation error. Result error can be viewed as the approximation error measured in result space. Result error nature and quantification is operation-dependent. In visualization, *image error* measures the color pixel difference of the image obtained using the simplified representation compared to that obtained with the accurate representation. Geometric queries with boolean output, such as collision detection, may report incorrect answers for some inputs when using simplified representations. In this case, result error depends on the size of the set of inputs for which the test reports incorrect answers, called *range error*. Similarly, reverberation sound operations measure the *sound wave error*, and geometric computations such as volume and area computation measure the error of the scalar value returned by the query, called *scalar error*.

The error perceived by the user is the result error and not the approximation error, so both the reduction parameter and representation selection in LOD-based processing should be based on result error and not on approximation error. However, there are three handicaps in using result error for such purpose. First, result error measurement is operation-dependent; second, actual result error values are context-dependent, e.g. image error depends on lighting model, point of view, etc. Finally, in most applications result error is hard to evaluate. Due to this problems, result error is roughly estimated using the approximation error.

5.1.1 Metrics for approximation error

Metrics used for the approximation error are critical because the approximation error is used both during simplification process and during LOD selection to estimate the result error.

The most relevant scheme for the evaluation of the approximation error between two surface representations S and S' has two components: a *distance function* and a *norm*. The distance function is defined on S, on S' or on both surfaces. The approximation error is evaluated with any function norm such as L_2 or L_∞.

The most useful metric for comparing two surfaces is the Hausdorff distance [Grü67]. The Hausdorff distance d_H of a surface S with respect to S' is defined as

$$d_H(S, S') = max_S(min_{S'}(dist(p, p'))), \tag{1}$$

where $dist$ is the Euclidean distance between two points. The symmetric Hausdorff distance d_{SH} is defined as

$$d_{SH}(S, S') = max(d_H(S, S'), d_H(S', S)). \tag{2}$$

5.1.2 Error components

3D models contain geometric and surface properties for visualization realism. Sometimes surface properties are defined globally, i.e. in a per-object basis, and sometimes are defined locally, in a per-face, per-vertex o per-corner basis.

Ideally, a metric for the approximation error must be good enough to ensure that if the approximation error is bounded in the sense defined by the metric then the result error is also reasonably bounded. Approximation error metrics for bounding range and scalar errors can be based only on geometric properties. Unfortunately, geometric approximation error is not enough to reasonably bound image error. In addition to the geometric error, 3D model simplification for visualization applications must deal with the error produced by modifying appearance attributes in the simplified representations, so the approximation error has several components: the geometric error, and an error for each visual property defined.

5.2 Measuring model complexity

Polyhedral representations have two kinds of complexity: *geometric complexity*, which depends on the number of faces, loops, edges and vertices of the model, and *topology complexity*, which depends on topology invariants of the represented surface such as the genus and the number of shells. Geometric and topology complexity are closely related by the Euler-Poincare formula

$$v - e + f = 2(s - g) + h, \tag{3}$$

where v, e, f, s, g and h are resp. the number of vertices, edges, faces, shells, genus and rings (interior loops in faces).

An application-independent approach for measuring geometric complexity is based on sum of the geometric entities of the polyhedral model, $f + e + v$, where f, e and v are resp. the number of faces, edges and vertices.

The application-dependent version of the previous index is defined as:

$$fc_f + ec_e + vc_v, \tag{4}$$

where c_f, c_e and c_v are resp. the application-dependent unit cost of processing a face, edge or vertex. For evaluation of storage requirements, c_f, c_e and c_v are resp. the number of bits required for storing a face, edge or vertex. Note that in some applications only one of these magnitudes is relevant. For example, backface culling cost depends only on the number of faces; wire-frame visualization depends on the number of edges, and per-vertex lighting cost depends only on the number of vertices.

Since geometric magnitudes are related by Euler's formula, some of them can be computed from the others. In the particular case of triangle meshes representing closed manifolds, any of f, e, v is enough for computing the others, since $3f = 2e$ [PRS89], assuming one shell and genus zero.

When comparing geometric complexity of two polyhedral representations, several metrics have been proposed. The most simple approach is:

$$f - f', \tag{5}$$

where f and f' are the number of faces of the input and output representations, resp. This is the most used metric for measuring complexity of triangle meshes [RB93], although it is not suitable for arbitrary polyhedral models since it ignores face complexity.

A face ratio does not depend on the absolute size of the model ([SZL92]):

$$\frac{f}{f'}. \tag{6}$$

We propose a new metric for measuring geometric complexity of arbitrary polyhedral models. This metric is required for comparing simplification methods generating triangle meshes with methods generating arbitrary polyhedral models:

$$\frac{f+e+v}{f'+e'+v'}. \tag{7}$$

5.3 Topology preservation vs. topology simplification

Geometric simplification deals with simplification of the geometric complexity of polyhedral models. *Topology simplification* deals with simplification of the topology complexity of polyhedral models. Since in polyhedral models topology is implicitly represented by geometric symbols, by topology simplification we mean the appropriate modification of the surface topology in order to further reduce geometric complexity. Note that topology simplification often involves genus and shell reduction, but in some cases involves genus or shell creation.

Topology preservation is required in some applications such as medicine visualization and molecular modeling, where presence or absence of tunnels is critical for correct data interpretation, specially if the simplified simplification is intended to substitute the input representation. However, topology preservation limits the amount of symbol reduction [Hea96] specially in topologically complex objects and assemblies. Topology simplification is suitable whenever surface topology has not essential meaning or when the simplified representation is intended to be part of a multiresolution model.

5.4 Curve Simplification

Curve simplification deals with simplification of polygonal curves embedded in 2D or 3D space. Curve simplification is a well known problem. There exists many different approaches and plenty of methods (see [HI86], [II88], [Vel92], [ET92], [Bal81]).

Polygonal curves are represented by an ordered sequence $p_1, \ldots, p_n$ of points called vertices. The vertices of the simplified curve are usually a subset of the original vertices, so the simplified curve can be represented by $p_{i_1}, \ldots, p_{i_m}$, where $1 = i_1 < i_2 < \ldots < i_m = n$.

Metrics to mesure the approximation error between curves are quite simple. All metrics are based on comparing each edge p_j, p_k of the simplified curve with the edges $p_j, \ldots, p_k$ being represented by this edge (see Figure 8).

The most common error metrics for curve simplification are defined as follows [II88] (see Figure 9):

- *e1 (distance edge-vertices)*
  $e1 = max_l(dist(segment(p_j, p_k), p_l)$, where l ranges from j to k.
- *e2 (distance straight line-vertices or parallel strip error)*
  $e2 = max_l(dist(line(p_j, p_k), p_l)$, where l ranges from j to k.

- *e3 (area minimum rectangle)*

  $e3 = area(rect(p_j, p_k))$, where $rect(p_j, p_k)$ is defined as the minimum rectangle oriented along the edge p_j, p_k containing all vertices ranging from j to k.

- *e4 (minimum side of minimum rectangle)*

  $e4$ =minimum size of $rect(p_j, p_k)$, where $rect$ is defined as in the previous metric. The minimum size of a rectangle is the minimum distance between two parallel edges.

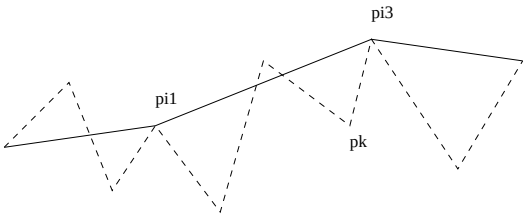


Figure 8: Input polygonal *(dashed)* and simplifed representation

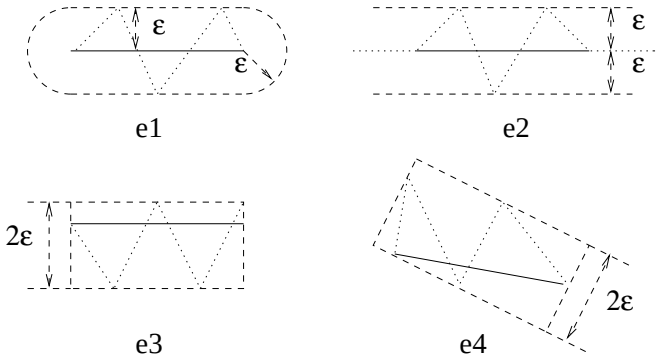


Figure 9: Error metrics for polygonal curves.

6 Characterization of surface simplification methods

In this section a new characterization of surface simplification methods is proposed. These criteria are grouped according whether they refer to input domain, approximation error, output domain, or algorithm internals.

6.1 Criteria about input domain

- *representation scheme domain*

  This work focuses on simplification of solids and surfaces represented by polygonal models. However, some surface simplification methods accept as input other kinds of representation schemes in addition to boundary representations, such as constructive models or space decomposition models.

- *geometric domain*

  Some simplification algorithms require the input representation to have particular geometric properties, such as convex surface or orthogonal faces. Other methods require two-manifold, non self-intersecting surfaces. This criterion includes pure geometric properties and the incidence properties defined by the topology of vertices, edges and faces.

- *topology domain*

  Some simplification algorithms require the input representation to describe a surface with particular topology, such as genus zero or single shell surfaces. Note that in this criterion we are interested in the topology of surfaces, not in the topology of geometric elements (vertices, edges, faces).

- *focus domain*

  Focus domain is the set of input representations for which the simplification algorithm is oriented, such as reconstructed surfaces.

- *scalability*

  Scalability measures how easy is to apply the simplification algorithm to large amounts of objects. Need of human intervention and non-intuitive user-defined parameters decrease scalability.

- *appearance preservation*

  This criterion indicates how the simplification algorithm handles visual information such as color and textures. There are three approaches: no support at all (visual info is discarded), partial support (visual info is not taken into account during the simplification process, but it is included in the simplified representation) and full support (the simplification process takes as input both geometric and visual information, both are simplified and present in the simplified representation). Partial support usually involves no vertex creation, so texture coordinates, vertex normals, etc. can be directly imported from the input representation.

6.2 Criteria about approximation error

- *user-defined parameters*

  This criterion is concerned with user-defined parameters required for the simplification algorithm.

- *reduction parameter specification*

  This criterion refers to how the reduction parameter is defined. The reduction parameter is user-defined and indicates in some way the desired degree of simplification. There are three ways of describing the reduction parameter: min_ϵ, $min_\#$ and weighting. In the min_ε approach, the reduction parameter gives the size desired for the simplified representation, by means of the number or percentage of vertices, edges or faces, and the simplification algorithm tries to generate a representation with required size with the minimum approximation error. In the $min_\#$ approach, the reduction parameter gives the maximum approximation error acceptable, and the simplification algorithm tries to generate a representation within this bound with the minimum number of geometric entities. In the weighting approach, the reduction parameter is a weight for the opposite goals of accuracy and concision.

- *error metric*

  The approximation error can be measured in many ways, such as the Hausdorff distance between two surfaces.

- *bounded error*

  This criterion indicates whether the user can define the maximum approximation error (using the $min_\#$ approach) or not.

- *error report*

  This criterion indicates whether the simplification algorithm allows the approximation error to be reported after simplification.

6.3 Criteria about output representations

- *representation scheme output*

  This criterion refers to the representation scheme of the output: a particular BRep (polyhedral model, polygon mesh or triangle mesh), or multiresolution model (collection of LODs, hierarchical or incremental). Some methods generate compressed BRep, such as triangle strips to represent triangle meshes.

- *geometric properties*

  Geometric properties of the simplified representations. For instance, preservation of relevant features such as sharp edges.

- *vertex creation*

  This criterion indicates whether the simplification algorithm generates vertices at new positions, or vertices of simplified representations are subsets of input vertices.

- *topology preservation*

  This criterion indicates whether the simplified representation represents a surface topologically equivalent to the input surface, or the simplification algorithm is able to change surface topology. Surface topology is determined by topological invariants such as genus and number of shells.

- *smooth transition*

  This criterion indicates whether the simplification algorithm provides some mechanism for smooth transition between consecutive LOD's, in order to avoid the abrupt change problem.

- *simplification limit*

  This criterion indicates the simplified representation generated by the simplification algorithm when maximum face reduction is demanded.

7 Surface simplification strategies

Surface simplification methods can proceed top-down or bottom-up. In *top-down* methods the key ingredient is a face reduction process. Top-down methods build a working representation initially set to the input representation, and reduce the number of faces by some face reduction strategy, so accurate representations are produced first than coarse ones, because coarse representations require more face reduction steps than accurate ones. In *bottom-up* methods the key ingredient is a refinement process. Bottom-up methods start from a very rough approximation and increase the number of faces by some refinement strategy, so coarse representations are produced first than accurate ones, because accurate representations require more refinement steps than coarse ones.

7.1 Face reduction strategies

A classification of face reduction strategies depends on the internal representation adopted. There are basically two approaches: those using directly a polyhedral BRep and those using a SDM.

Face reducion strategies based on polyhedral BReps are:

- *vertex clustering*

  The vertex clustering strategy consists of grouping close vertices in groups called *clusters*. All vertices inside a cluster are replaced by a single vertex, called cluster representant. These replacements cause many edges and faces to collapse. Repeated collapsed entities are removed. This strategy is adopted in [RB93], [Tan97] and [Red96].

- *incremental face reduction*

  Incremental face reduction (also known as decimation) works by the iterative elimination of geometric entities through a *local reduction operator* chosen upon local geometric optimality criteria. Local reduction operators are the basis of the majority of simplification methods [SZL92], [Gue96], [RR96], [AS96], [GH97], [Ham94], [DZ91], [KCHN91], [HH93], [KT96] , [KT93], [CCMS97], [BBCS96], [HDD+93], [Hop96], [Hop97] and [PH97].

- *re-tiling*

  The re-tiling strategy consists of introducing new vertices on the original representation which are moved over maximal curvature locations; then a new triangulation is built including the original and the new vertices; finally, original vertices are removed by a local reduction operator. This strategy was originally proposed in [Tur92].

- *wavelet decomposition*

  Simplification based on wavelet decomposition proceeds through two steps: re-meshing, where a simple mesh approximating the input surface is generated, and wavelet parameterization. Wavelet decomposition is used in several simplification methods [Eea95], [DLW94], [CPD+96] and [GSG96].

Face reduction strategies based on SDM first convert the input representation into a SDM; a polyhedral surface is then constructed from the SDM. Accuracy level is determined by SDM cell size.

7.2 Local operators over triangle meshes

In this section we compare the mesh operators, which are the key ingredient of the majority of surface simplification methods, specially those following an incremental face reduction strategy. Local operators either reduce mesh complexity (reduction operators) or improve the fitting of the simplified mesh (fitting operators).

The following reduction operators have been proposed in the geometry simplification literature (see Figure 10):

- *vertex-removal*

  The vertex-removal operator takes as parameter the vertex to be removed. The vertex and its t incident triangles are removed. The resulting hole is triangulated using $t-2$ triangles. The only computed parameter is the new incidence graph of the hole's triangulation, since no new vertices are inserted. The operator reduces the overall number of faces by two, the number of edges by three and the number of vertices by one. Except in degenerate cases which can be easily identified by inspection of the incidence graph, the vertex-removal operator preserves the topology of the mesh. To avoid self-intersections, additional geometric tests are required. The selection of the vertex to be removed is commonly based on a curvature estimation at the vertex. Vertices with low curvature values are removed first. The vertex removal operator is used in [SZL92], [ea96], [SL96] and in most terrain simplification methods.

- *edge-collapse*

  The edge-collapse operator takes as parameter the edge to be collapsed, or its equivalent, a pair of vertices sharing an edge. The two vertices are collapsed in one vertex. As a result of this collapse, the triangles sharing the edge degenerate in a segment and are removed. The only computed parameter is the new vertex position, which usually is that of one of the two old vertices, or a weighted average. The operator reduces the number of faces by two, the number of edges by three and the number of vertices by one. Except in degenerate cases which can be easily identified by inspection of the incidence graph, the edge-collapse operator preserves the topology of the mesh. To avoid self-intersections, additional geometric tests are required. The selection of the edge to be removed is commonly based on an estimation of the error in the Hausdorff distance sense. Usually, feasible edge-collapse operators are computed in a previous step, and stored in priority queue ordered by error. The edge-collapse operator is used in most of the state-of-the-art methods: [RR96], [Gue96], [AS96], [Hop96], [Hop97].

- *vertex-clustering*

  The vertex-clustering of v vertices is conceptually equivalent to $v-1$ vertex-clustering operation involving just two vertices (pair contraction), so we will review only this latter form. The pair contraction operator takes as parameter the two vertices to be collapsed. When these vertices are connected along an edge, this operation is an edge-collapse, but disconnected vertices are also allowed to be collapsed. If the latter case, only the number of vertices is reduced; otherwise, the triangles sharing the edge degenerate in a segment and are removed. The only computed parameter is the new vertex position, which usually is that of one of the two old vertices, or a weighted average. If the vertices are connected by an edge, the operator reduces the number of faces by two, the number of edges by three and the number of vertices by one; otherwise only reduces the number of vertices. Unlike previous reduction operators, the pair contraction does not preserve the topology of the mesh, and creates non-manifold meshes. To avoid self-intersections, additional geometric tests are required. In the

simpler methods, the selection of the vertices to be collapsed is based on geometric proximity. Usually, several vertices are clustered at a time. The vertex-clustering operator is used in [RB93], [RR96], [Red96], [GH97].

- *face-removal*

  The face-removal takes as parameter the triangle T to be removed. This triangle and all its neighbors sharing one vertex with T are removed. The resulting hole is triangulated with the help of a new vertex. The computed parameters are the coordinates of the new vertex, and the incidence graph of the triangulation. The operator reduces the number of faces by four, the number of edges by six and the number of vertices by two. Except in degenerate cases which can be identified [Ham94], the face-removal operator preserves the topology of the mesh. Due to the high number of triangles involved, incremental methods using face-removal rapidly arrive to a mesh which cannot be further simplified while maintaining its validity, stopping at representations where other simpler operators such as edge-collapse could be applied. The face removal operator is used in [Ham94].

In addition to reduction operators, which modify the count of the geometric entities of the mesh, surface simplification methods often rely on fitting operators:

- *edge-flip*

  The edge-flip operator (also known as edge-swap) takes as parameter two adjacent triangles (or their shared edge). The non-planar quadrilateral resulting of merging together both triangles is triangulated using the opposite diagonal. There are no computed parameters. The edge-flip operator preserves the topology of the mesh but not the incidence graph. To avoid self-intersections, additional geometric tests are required. The edge-flip is used for two purposes: a) to improve the fitting of the simplification to the original surface, in non-flat regions, and b) to create well-shaped triangles, in flat-regions. The edge-flip operator is used in [Hop96], [CCMS97], [BBCS96], to minimize the error produced by another previous operator, such as edge-collapse.

- *vertex-displacement*

  The vertex-displacement operator take as parameter the vertex to be moved. The only computed parameters are the new coordinates, usually given as an offset vector. The vertex-displacement operator preserves the topology of the mesh and the incidence graph. To avoid self-intersections, additional geometric tests are required. The vertex-displacement is used to locally improve the fitting of the simplification to the original surface. Its relevance is due to the fact that the vertex displacement is closed in the domain of valid triangle meshes (i.e. it preserves the validity); this does not hold for arbitrary polyhedra, since vertex-displacement would produce non-planar faces.

In our opinion edge-collapse is the most suitable operator for face reduction. There are two reasons for that: its simplicity (only affecting two triangles at a time), allowing further simplification when other operators cannot be applied, and its generality, in the sense that all topology-preserving reduction operators can be derived from a series of edge-collapses and edge-flips:

- One vertex-removal can be replaced by an edge-collapse and several edge-flips.
- The multiple vertex-clustering of n vertices can be replaced by $n-1$ vertex-clustering of two vertices, which in turn is a generalization of edge-collapse to arbitrary pairs of vertices.

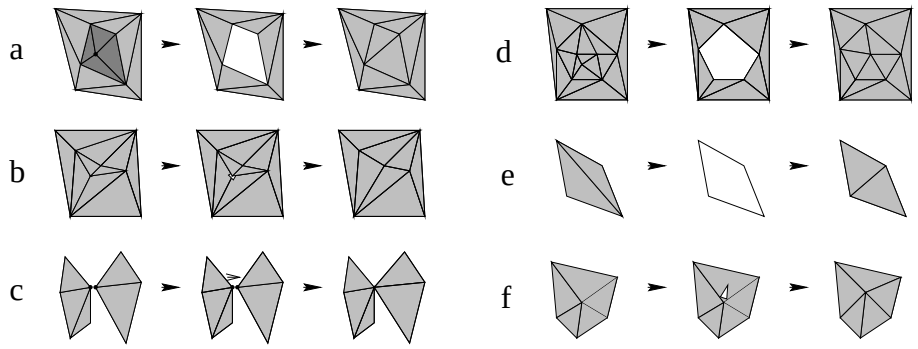


Figure 10: Reduction and fitting operators: a) vertex-removal, b) edge-collapse, c) vertex-clustering, d) face-removal, e) edge-flip and f) vertex-displacement.

- One face-removal can be replaced by two vertex removal, and hence by two edge-collapses and several edge-flips. Note that, in the face-removal we are considering, the triangulation is performed introducing a new vertex. Otherwise, three edge-collapses instead of two would be necessary.

8 Review of simplification methods

8.1 Classification of surface simplification algorithms

Here we propose a new classification of surface simplification methods. Recall that non polyhedral surface simplification, and loss-less compression are out the scope of this work.

A first group of simplification methods are terrain simplification methods. Terrain models have special properties that are taken into account by terrain simplification methods: simple surface topology, they can be represented using elevation grids, error measurement is simple, etc.

The second group is formed by triangle mesh simplification methods (mesh simplification for short). Mesh simplification methods are classified according to the face reduction strategy: vertex clustering, incremental face reduction, re-tiling or wavelet decomposition.

The third main group includes all methods based on space decomposition models.

- SS
  - Terrain [SR94]
  - Mesh simplif
    - Bottom-up: [Vel93]
    - Clustering: [RB93], [Red96], [Tan97], [GH97]
    - Incremental
      - vertex: [SZL92], [ea96], [SL96]
      - edge: [Gue96], [RR96], [AS96], [HDD+93], [Hop96], [Hop97], [PH97]
      - face: [Ham94],
      - faces: [DZ91], [KCHN91], [HH93], [KT96] , [KT93]
      - edge-flip: [CCMS97], [BBCS96]
    - Re-tiling: [Tur92]
    - Wavelets: [DLW94], [Eea95], [CPD+96], [GSG96]
  - Based on SDM
    - Voxels: [HHK+95], [Hea96], [MSS94], [HB94]
    - Octrees: [JAS95], [AAB95a], [AAB+96a]

8.2 Terrain simplification methods

Terrain simplification deals with simplification of digital terrain models (DTM for short). DTM models are broadly used in Geometric Information Systems (GIS for short). The most common representations for DTM are:

- *points on a regular grid*

  The array of height values is imported together with the grid information (resolution, location of the sampled area, etc).

- *scattered points*

  Scattered sampling poings can be imported as a list of coordinates together with height values.

- *Triangulated Irregular Network (TIN)*

  Sampling points are read in together with their incidence information, forming a triangle mesh.

- *contour lines*

  Contour lines are lists of coordinates associated with a certain isovalue.

Terrain simplification is a well known problem, and many methods have been proposed in the literature. See [GH95] for a review of them; in this section we will discuss a representative of the family of terrain simplification methods. Since terrain simplification methods take profit of particular properties of terrain models, they are not effectively applicable to arbitrary polyhedral models.

8.2.1 Terrain simplification using vertex-removal (Schroder)

A terrain simplification method is presented in [SR94]. This method follows a top-down approach based on a TIN representation of the terrain. The face reduction strategy adopted is an incremental reduction based on the vertex-removal operator. The decimation criterion is based on an estimation of each vertex contribution to the overall shape. Given a vertex v shared by a set T of triangles, the contribution a_{max} of v is computed as the maximum angle between normal vectors of triangles in T and the weighted average normal vector. (see Figure 11).

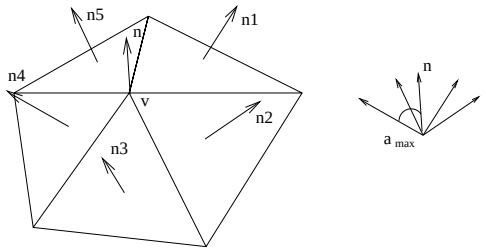


Figure 11: Contribution of a vertex

The algorithm admits two operation modes, corresponding to the min_ϵ and $min_\#$ goals. In the first case, a user-defined parameter sets the desired number or vertices v'. Vertices are sorted according to their contribution and the $v - v'$ less significant vertices are removed. In the second case, the user defines a threshold consisting of the maximum angle A_{max} of a vertex to be removed, and all vertices whose contribution is less than A_{max} are removed.

To avoid the removal of smooth accidents, both the original contribution and the current contribution of a vertex are maintained. A vertex is removed only when both contribution values are less than A_{max}.

The following tables summarize the main properties of the method:

Terrain simplification using vertex-removal (Schroder)	
Input	
Representation scheme	Triangle mesh
Geometric domain	Terrain model
Topology domain	Unrestricted
Focus domain	Terrain models
Appearance preservation	Color can be included in decimation criteria
Approximation error	
User-defined parameters	Maximum angle A_{max} or target number of vertices
Reduction parameter	Maximum angle A_{max} or target number of vertices
Error measurement	Local
Bounded error	No
Error report	No
Output	
Output Scheme	Triangle mesh
Geometric properties	Preserves even very small sharp edges
Vertex creation	No
Topology preservation	No
Simplification limit	Mesh with no triangles
Usability	Good
Topology tests	None
Smooth transition	No

The main contribution of this work with respect to the method proposed in [SZL92] is the new criterion for vertex removal, which takes into account both the original contribution and the current contribution after some simplification steps.

8.3 Mesh simplification methods

Mesh simplification methods work with triangle meshes. To simplify arbitrary polyhedral models, a prior triangulation of the faces is required. In addition, these models produce simplified representations involving only triangular faces.

As far as we know, all top-down methods internally using BRep representations always deal with triangle meshes instead of arbitrary polyhedral models. There are two main reasons for that: on one hand, in many cases the input representation is triangulated (as triangulated is the output of most surface reconstruction methods) or needs to be triangulated (for visualization purposes); on the other hand, incidence relations of triangle meshes are far simple than arbitrary polyhedra [And98b]. The first outcome of this fact is that reduction operators become simpler. For instance, we can modify the coordinates of a vertex in a triangle mesh without producing a non-planar face, fact that does not hold on non-triangular models.

In this section we review the most relevant mesh simplification methods [RB93], [SZL92], [ea96], [Ham94], [Tur92], [HDD+93], [Eea95], [DLW94], [CPD+96] and [KT93], although other methods have been proposed [Hop96], [BBCS96], [KS96], [DZ91], [GGS95].

8.3.1 Vertex-clustering (Rossignac, Borrel)

A simple yet efficient visualization-oriented mesh simplification method is proposed in [RB93]. The face reduction strategy is based on all-at-once vertex-clustering. The method proceeds through five steps:

- *weigth computation*

  For each vertex, a weight measuring its perceptual importance is computed. The weight computation is based on the maximum angle and lengths of edges inciding at the vertex.

- *triangulation*

  If the input representation is not a triangle mesh, faces are triangulated.

- *vertex grouping*

  Nearby vertices are grouped into clusters through gridding or coordinate truncation. The reduction parameter is defined by the grid subdivision.

- *synthesis*

  Each vertex cluster is unified in a single vertex, computed as the weithed average of the vertices inside the cluster (for a smoothing effect) or just the vertex with maximum weight (for detail removal).

- *elimination*

  Vertex clustering causes many triangles to collapse into edges or points. In this step, repeated collapsed entities are removed. The result representation involves triangles, segments and points as primitives.

The main advantage of this method is that it is very simple and efficient, and that the reduction parameter is very intuitive. Vertex positions can be parameterized so transition between consecutive LOD levels is smooth.

Since this method is visualization-oriented, model integrity is very relaxed and the method produces simplified representations with dangling triangles, segments and points.

Vertex-clustering (Rossignac, Borrel)	
Input	
Representation scheme	Polyhedral model
Geometric domain	Unrestricted
Topology domain	Unrestricted
Focus domain	Visualization
Appearance preservation	No
Approximation error	
User-defined parameters	Size or number of clusters
Reduction parameter	Size or number of clusters
Error measurement	Hausdorff distance between meshes
Bounded error	No
Error report	Yes
Output	
Output Scheme	List of triangles, segments and points
Geometric properties	Preserves sharp edges depending on synthesis
Vertex creation	Depends on synthesis
Topology preservation	No
Simplification limit	An isolate point
Usability	Good
Topology tests	None
Smooth transition	Yes (vertex parameterization)

8.3.2 Decimation of triangle meshes (Schroeder et al.)

The mesh simplification method presented in [SZL92] follows a top-down approach. The face reduction strategy is based on incremental simplification based on the vertex-removal operator. The algorithm makes multiple passes using local geometry and incidence to remove vertices that satisfy a distance or angle criterion. During a pass, all vertices that meet the specified criteria are removed. The three steps of the algorithm are:

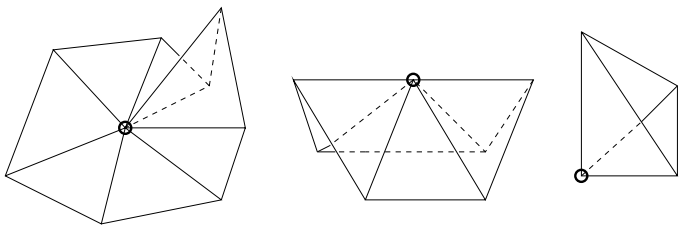


Figure 12: Vertices with one, two and three feature edges, resp.

- *characterize the local vertex geometry and incidence*

  Each vertex may be assigned one of five vertex configurations (see Figure 12). A *boundary vertex* is a vertex that is on the boundary of the mesh. Non-boundary vertices are classified as manifold and non-manifold (called *complex*) vertices. Manifold vertices can be further classified depending on the number of feature edges, i.e. edges whose inciding triangles have a dihedral angle greater than a user-defined *feature angle*. A

vertex without feature edges is a *simple* vertex. A vertex used by two feature edges is an *interior edge* vertex. If one, three or more feature edges use the vertex, the vertex is classified as *corner* vertex.

- *evaluate the decimation criteria*

  Vertex evaluation depends on its classification and on user-defined parameters. A user-defined parameter determines whether feature edges must be preserved or not. Boundary vertices are evaluated according to an edge-vertex distance. Complex vertices are not deleted from the mesh. Simple vertices are evaluated measuring the distance to an average plane. When feature edge preservation is selected, corner and interior edge vertices are evaluated according to an edge-vertex distance. Otherwise, the distance to an average plane is used instead.

- *apply the vertex-removal operator*

  The vertex and its associated triangles are removed. The resulting hole, which is star-shaped, is triangulated using a recursive splitting algorithm. Several tests are made in order to preserve the surface topology.

Unfortunately, the local error is accumulated through many iterations so the final approximation error is not bounded.

Decimation of triangle meshes (Schroeder et al.)	
Input	
Representation scheme	Triangle mesh
Geometric domain	Unrestricted
Topology domain	Unrestricted
Focus domain	Unrestricted
Appearance preservation	Partial (no vertex creation)
Approximation error	
User-defined parameters	Reduction percentage or target number of vertices; edge preservation (boolean); feature angle; maximum distance
Reduction parameter	Reduction percentage or target number of vertices
Error measurement	Local
Bounded error	No
Error report	No
Output	
Output Scheme	Triangle mesh
Geometric properties	Smoothes or preserves feature edges depending on parameters
Vertex creation	No
Topology preservation	Yes
Simplification limit	Tetrahedron (for genus 0 meshes)
Usability	Good
Topology tests	Many tests after triangulation
Smooth transition	No

8.3.3 Simplification envelopes (Cohen et al.)

A general framework for a family of mesh simplification methods is presented in [ea96].

Simplification envelopes, are a generalization of *offset surfaces* that cover both sides of the input surface. Face reduction is made inside the volume enclosed by the envelopes. Approximation error is not uniform but it is bounded. Envelopes provide a visual mechanism for adapting the approximation error through the input surface in a region basis, according to its perceptual importance. However, such adaptation requires human intervention.

The framework is oriented to incremental mesh simplification methods using the vertex-removal operator with the required tests for preserving the surface topology.

Simplification Envelopes (Cohen et al.)	
Input	
Representation scheme	Triangle meshes
Geometric domain	Two-manifold
Topology domain	Unrestricted
Focus domain	Unrestricted
Appearance preservation	Partial (no vertex creation)
Approximation error	
User-defined parameters	Tolerance ϵ
Reduction parameter	ϵ
Error measurement	Hausdorff distance
Bounded error	Yes
Error report	Yes
Output	
Output Scheme	Triangle mesh
Geometric properties	Depends on decimation criteria
Vertex creation	No
Topology preservation	Yes
Simplification limit	Tetrahedron (for genus 0 meshes)
Usability	Adaptation of tolerance requires human intervention
Topology tests	Several tests before vertex-removal
Smooth transition	No

8.3.4 Simplification through face removal (Hamman)

Another triangle mesh simplification method is presented in [Ham94]. The method follows a top-down approach. Unlike other incremental algorithms [SZL92], [Tur92], the face reduction strategy adopted is based on the face-removal operator (see Figure 10).

The input representation is a two-manifold triangle mesh. The three steps of the algorithm are:

- *compute weigths of triangles*

  For each triangle of the input mesh, a weigth is computed using the main curvature [Far90] at its vertices (to preserve triangles at high curvature regions) and its interior angles (to preserve good aspect ratio of triangles). Triangles are sorted according to their weigths.

- *elimination test*

  In each iteration, a test is evaluated to determine if a face removal operator can be

applied to the triangle with minimum weigth without changing mesh topology. Given a triangle T, its *corona* is the set of triangles sharing some vertex with T, excluding T itself (see Figure 13 a). A corona is called *connected* if it can be cyclically ordered so that each pair of consecutive triangles share an edge (see Figure 13 b). A corona is *cyclic* if it contains a subset of cardinality three where each triangle shares an edge with two triangles (see Figure 13 c y d). Only triangles whose coronas are connected and are cyclic can be removed. An additional halfspace criterion test guarantees that resulting holes can be triangulated preserving the mesh topology.

- *face removal*

  If all decimation criteria are meet, the triangle and its corona are removed. The resulting hole is triangulated inserting a new vertex (see Figure 13 e). Then, some edge-split operators are applied over the new triangulation in order to improve the aspect ratio of new triangles (see Figure 13 f). Finally, a weigth is computed for each new triangle.

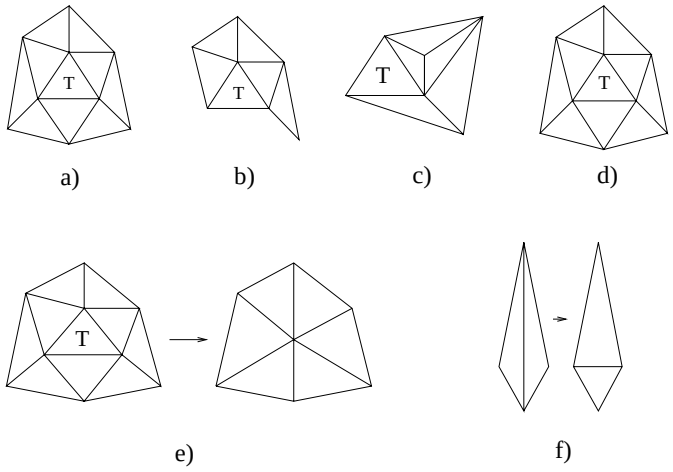


Figure 13: a) A triangle and its corona; b) disconnected corona c) cyclic corona; d) acyclic corona; e) triangulation; f) diagonal interchange

The incremental simplification stops when some user-defined reduction ratio is reached or when no more triangles can be deleted without changing mesh topology.

Since decimation criteria is based on curvature, and local errors are accumulated, the global approximation error is not bounded. Decimation can stop prematurely in still complex meshes due to the large number of triangles and tests involved in decimation criteria.

Simplification through face removal (Hamman)	
Input	
Representation scheme	Triangle mesh
Geometric domain	Two-manifold
Topology domain	Unrestricted
Focus domain	Tessellations of polynomial surfaces
Appearance preservation	No
Approximation error	
User-defined parameters	Reduction percentage
Reduction parameter	Reduction percentage
Error measurement	Local
Bounded error	No
Error report	No
Output	
Output Scheme	Triangle mesh
Geometric properties	Two-manifold; good aspect ratio
Vertex creation	Yes
Topology preservation	Yes
Simplification limit	Often is unable to reduce up to a tetrahedron
Usability	Good
Topology tests	Connected, cyclic and halfspace tests in each iteration
Smooth transition	No

8.3.5 Superfaces simplification (Kalvin, Taylor)

A method for polyhedra simplification using superfaces is presented in [KT93].

The three steps of the algorithm are:

- *superface creation*

  A *superface* is the non-planar contour of a set of connected faces of the input polyhedron. Superface creation is a greedy process of grouping adjacent faces (no geometric changes are introduced yet). Grouping criteria involve a planarity test (the distance between superface vertices and a set of planes approximated by an ellipsoid must be less than some user-defined D_{max}), an orientation test (the angle between the face being inserted in the superface and an average plane must be less than an user-defined Ω_{max}), a test for avoiding fold-over of the surface and an aspect ratio test computed as $perimeter^2/area$. Superface grouping stops when every input face belongs to a superface.

- *border straightening*

  The countour of each superface is divided into segments which are simplified with a 2D polygonal simplification method. The maximum deviation allowed for the 2D simplification is computed from D_{max} and Ω_{max}.

- *superface triangulation*

  Since superfaces are non-planar contours, they must be triangulated. Its vertices are projected into an average plane and the superface is divided in star-shaped polygons, which are independently triangulated.

Although the input representation might contain arbitrary faces, the method only produces triangles. Since planarity, orientation and fold-over tests does not take into account face area, non-planar regions cannot be simplified even when involved faces are small. Unfortunately, the simplified mesh can self-intersect when the input does not, since two triangles from different superfaces may intersect.

Superfaces simplification (Kalvin, Taylor)	
Input	
Representation scheme	Polyhedral model
Geometric domain	Unrestricted
Topology domain	Unrestricted
Focus domain	Local
Appearance preservation	Partial (no vertex creation)
Approximation error	
User-defined parameters	D_{max}, Ω_{max}, d_{max}
Reduction parameter	D_{max}, d_{max}, Ω_{max}
Error measurement	Distance between input vertices and simplied mesh
Bounded error	Yes: $2D_{max}$
Error report	Yes
Output	
Output Scheme	Triangle mesh
Geometric properties	Might produce selfintersections
Vertex creation	No
Topology preservation	Yes
Simplification limit	Tetrahedron (for genus 0 meshes)
Usability	Regular
Topology tests	No reported
Smooth transition	No

8.3.6 Mesh optimization (Hoppe et al.)

A mesh simplification using an optimization approach is presented in [HDD+93]. This method follows a top-down approach based on a triangle mesh representation. The face reduction strategy adopted is an incremental reduction based on four local operators: edge-colapse, edge-split, edge-flip and vertex displacement. Unlike other incremental approaches, whose face reduction is driven by local criteria, in the method proposed in [HDD+93] face reduction is directed by minimization of an energy function.

The energy function is defined over the domain of triangle meshes and models the opposite goals of concise representation and fidelity to the original mesh. Given the incidence graph K and the vertices V of the input mesh, the energy function is defined as the sum of three components:

$$E(K,V) = E_{dist}(K,V) + E_{rep}(K) + E_{spring}(K,V) \tag{8}$$

E_{dist} measures fidelity to the input mesh as the sum of squared distances of the points in X from the current mesh, where X is a set of points over the original mesh computed by uniform sampling: $E_{dist}(K,V) = \sum d^2(x_i, M)$. E_{rep} measures the concision of the

representation and is proportional to the number of vertices, $E_{rep}(K) = c_{rep}v$, where c_{rep} is a user-defined weight constant. E_{spring} is a somewhat artificial component for keeping the simplified mesh from deviate in regions with low density of samples in X, measured as the sum of edge lengths. The need of such component is a direct consequence of measuring the approximation error as point to surface distance, because displacement of mesh vertices without no sampled points close to them have no effect on E_{dist}. Unfortunately, E_{spring} has lateral effects as it penalties long edges. A user-defined constant, c_{spring} weights this component.

The optimization problem is decomposed into two nested subproblems, although the method does not guarantee the detection of the energy function global minimum:

- *vertex optimization*

  Vertex optimization deals with optimizing the energy function by changing vertex coordinates while preserving mesh incidence. Vertex optimization is based on the vertex displacement operator. First, for each point in X the closest point on the current mesh is computed. Then, vertex positions are changed by square minimization.

- *connectivity optimization*

  Connectivity optimization deals with optimizing the energy function by changing the incidence graph of the mesh through local operators while preserving vertex locations. Connectivity optimization is based on a reduction operator (edge-collapse), a refinement operator (edge-split) and a fitting operator (edge-flip) (see Figure 10). The edge-split is the inverse operation of an edge-collapse. The methods preserves surface topology since only legal (i.e. topology preserving) operators are applied.

The main advantage of this approach is the high quality of the simplified representation, consequence of the energy function minimization and of changing vertex positions. The algorithm can also be used for surface fitting to an irregular point set.

Unfortunately, long processing times are required. Since approximation error is measured using a set of sampled points, the user is required to provide a weight c_{spring} for the E_{spring} component. Since user-defined parameters are non-intuitive, unit-less weigth constants, they are difficult to set up (specially the reduction parameter c_{rep}) and results are hard to predict. The approximation error is unbounded.

Mesh optimization (Hoppe et al.)	
Input	
Representation scheme	Triangle mesh
Geometric domain	Two-manifold
Topology domain	Unrestricted
Focus domain	Unrestricted
Appearance preservation	Yes (in a subsequent work)
Approximation error	
User-defined parameters	C_{rep}, E_{spring} (unit-less) and number of samples
Reduction parameter	C_{rep}
Error measurement	distance between samples and simplified mesh
Bounded error	No, in the Hausdorff distance sense
Error report	E_{dist}
Output	
Output Scheme	Triangle meshes
Geometric properties	Depends on sampling. Might preserve feature edges
Vertex creation	Yes
Topology preservation	Yes
Simplification limit	Tetrahedron (for genus 0 meshes)
Usability	Poor (non-intuitive parameters)
Topology tests	Tests in each iteration
Smooth transition	Yes (in a subsequent work)

8.3.7 Re-tiling polygonal surfaces (Turk)

A mesh simplification method based on an original re-tiling approach is presented in [Tur92]. The method follows a top-down approach. The face reduction strategy is based on a vertex-removal operator.

The three steps of the algorithm are:

- *generate final vertices*

  The first step is to choose a set of points that will, at a later step, become the vertices of a new triangular tessellation of the surface. The number of points is user-defined and provides a mechanism for selecting the reduction degree. Once all points have been randomly placed on the surface, a relaxation procedure is applied to move each point away from its nearby points through evaluation of repulsion forces.

- *insert new vertices in the input mesh*

  The new vertices are inserted in the original polygons, and faces are triangulated using both the new and old vertices. This intermediate representation is called mutual tessellation, and it is topologically equivalent to the input surface.

- *remove old vertices*

  The method finally tries to remove old vertices using a vertex-removal operator. Old vertices are not removed when the operation changes the surface topology. Unlike the method proposed in [SZL92], topology preservation tests are made before the hole is triangulated. The result representation involves the new vertices and some old vertices.

The main advantage of this approach over other incremental methods based on the vertex-removal operator is that it is capable of creating vertices at new positions. Unfortunately, the approximation error is not bounded, and new vertices are placed without taking into account surface curvature, so the method does not preserve sharped edges and vertices.

Re-tiling polygonal surfaces (Turk)	
Input	
Representation scheme	Polyhedral model
Geometric domain	Two manifold
Topology domain	Unrestricted
Focus domain	Smooth surface
Appearance preservation	No
Approximation error	
User-defined parameters	Target number of vertices
Reduction parameter	Target number of vertices (the result might include more vertices)
Error measurement	No
Bounded error	No
Error report	No
Output	
Output Scheme	Triangle mesh
Geometric properties	Does not preserve feature edges
Vertex creation	Yes
Topology preservation	Yes
Simplification limit	Tetrahedron (for genus 0 meshes)
Usability	Good
Topology tests	Tests for all input vertices
Smooth transition	Yes

8.3.8 Multiresolution analysis (Eck at al.)

A wavelet-based mesh simplification method [You94], [RBC+92], [Sak95], [Dau92] is presented in [Eea95]. The method is a generalization of a previous multiresolution analysis method [DLW94] that overcomes the subdivision connectivity limitation. A triangle mesh has *subdivision connectivity* when it can be obtained from a simple mesh through the recursive 4-to-1 subdivision presented in Figure 14.

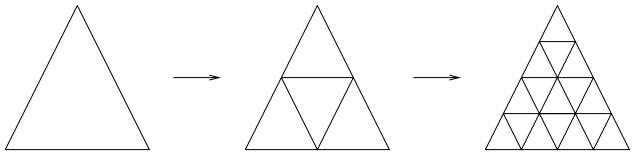


Figure 14: 4-to-1 subdivision connectivity

The two steps of the algorithm are:

- *re-meshing*

  The input mesh M is approximated by another mesh topologically equivalent but with subdivision connectivity. First, a very simple mesh is computed using Voronoi diagrams [PM85]. Then, a parameterization of M over this simple domain is computed. Finally, 4-to-1 subdivisions are applied over the simple domain, inserting new vertices until the Hausdorff distance between M_j and M is bounded by a user-defined ϵ_1.

- *wavelet parameterization*

  The second step is based on multiresolution analysis [DLW94]. The parameterization is decomposed into a low-resolution part and a sequence of local correction terms called wavelet coefficients. Decomposition tolerance is user-defined as ϵ_2.

Topology preservation is inherent to the method. The number of triangles of the original mesh is doubled in the approximated mesh. Since multiresolution analysis cannot be applied directly over the input mesh, the final multiresolution representation is lossy even at the higher level of the decomposition.

Multiresolution analysis (Eck at al.)	
Input	
Representation scheme	Triangle mesh
Geometric domain	Unrestricted
Topology domain	Unrestricted
Focus domain	Unrestricted
Appearance preservation	No
Approximation error	
User-defined parameters	Remeshing tolerance ϵ_1 and decomposition tolerance ϵ_2
Reduction parameter	ϵ_1 and ϵ_2
Error measurement	L_∞ of Hausdorff distance
Bounded error	Yes
Error report	Yes
Output	
Output Scheme	Multiresolution of triangle meshes
Geometric properties	No reported
Vertex creation	Yes
Topology preservation	Yes
Simplification limit	Domain mesh
Usability	Regular
Topology tests	No
Smooth transition	Yes

8.4 Methods based on space decomposition models

8.4.1 SDM-based simplification pattern

Polyhedral surface simplification using intermediate SDM was originally proposed in [AAB95a] and [HHK+95]. SDM-based simplification proceeds through three steps:

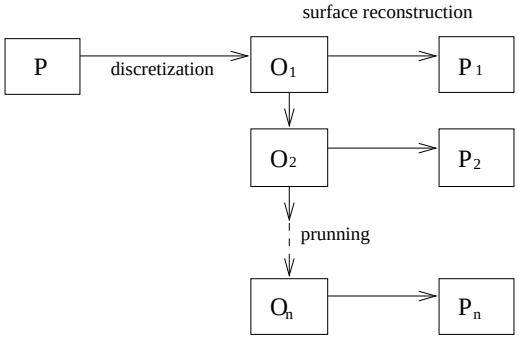


Figure 15: Example of simplification based on a SDM

- *BRep to SDM conversion*

  The first step is the generation of a SDM from the input polyhedral representation. According to the particular SDM adopted, current methods are based on voxel decompositions [Hea96], face octrees [JAS95] and classical octrees [AAB95a], [AAB+96a].

- *polyhedral surface reconstruction*

  This step involves the extraction of a polyhedral surface from the SDM. Some methods use an isosurface extraction approach, using binary values [JAS95], [AAB95a], [AAB+96a] or scalar values [Hea96], [HHK+95]. In the later case a isodensity value must be given.

- *face reduction*

  An optional face reduction process reduces the verbosity of the surface representations extracted in the previous step.

Since generation of multiresolution SDM is straightforward (it just requires glueing adjacent cells), the last two steps can be repeatedly applied over coarse representations of the SDM (see Figure 15), and hence produce a multiresolution of polyhedral representations. A multiresolution of octree representations can be obtained by iteratively pruning the last level of the octree.

8.4.2 Discretization and loss of data

The conversion of a BRep into a discrete representation (either a voxel decomposition, octree or 3D picture) can be viewed as a discretization function dis that takes a polyhedral representation and returns its discrete representation. Function dis naturally induces an equivalence relation in the polyhedra domain: two polyhedral representations P and P' are equivalent iff $dis(P) = dis(P')$. Equivalence classes give information about the data lost in the conversion.

In voxel decompositions, voxel represents the solid inside a cubic region of the space. White voxels correspond to regions completely outside the solid; black voxels correspond to regions containing a part of the solid. Two polyhedra P and P' are equivalent with respect to such discretization iff for each voxel c, $P \cap c = \emptyset \leftrightarrow P' \cap c = \emptyset$. Furthermore, if two polyhedra are equivalent then their symmetric difference is contained in the set of black voxels of their voxelization. The Hausdorff distance between the interior of two equivalent solids is less than the length of the main diagonal of a voxel (see Figure 16 b).

In 3D digital pictures, which represent solid objects through a regular grid of points, two polyhedra are equivalent iff for each lattice point p, $p \in P \leftrightarrow p \in P'$, and hence, the symmetric difference is not confined to any particular region and the Hausdorff distance is not reasonably bounded (see Figure 16 a).

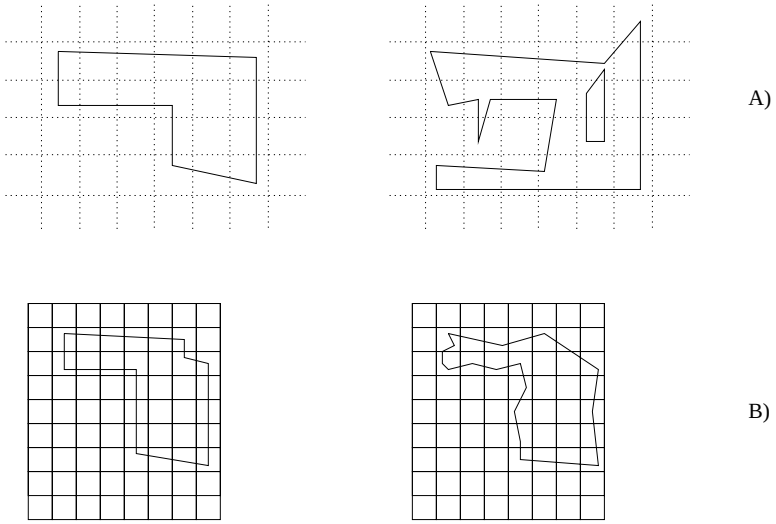


Figure 16: Equivalent polyhedra with respect to a 3D picture (a) and a voxel decomposition (b)

8.4.3 The surface reconstruction problem

Surface reconstruction from discrete data is a well known problem [Kal92], [PTN93], [GW94], [KR89]. In this section we review the main problems encountered in reconstructing polyhedral surfaces from octree and voxel-based representations, from the point of view of SDM-based simplification.

Isosurface extraction algorithms can be classified into two main groups [Kal92]: 2D algorithms and 3D algorithms. In 2D algorithms, volume data is processed as a series of 2D slices; surface reconstruction proceeds through two steps: first, a 2D contour is extracted from each slice, and then a surface is generated by merging contours of adjacent slices (see Figure 17). Contour-based surface reconstruction algorithms produce verbose representations because face size is limited by the distance between slices. Furthermore, contour matching often requires human intervention due to the ambiguity problem.

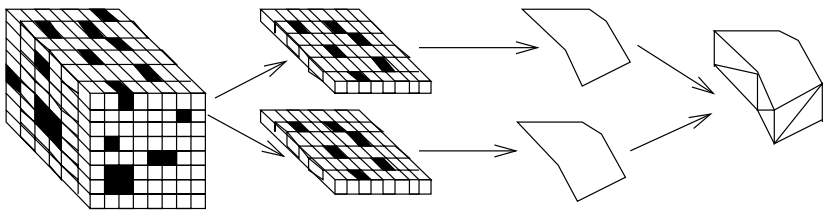


Figure 17: Surface reconstruction algorithms based on contours

In 3D algorithms, both the geometry and the incidence of the representation is generated directly from the SDM. There basically two families of such methods [Kal92]: *block-form* or *cuberille* methods [FLP89] and *beveled-form* of *marching cubes* methods. Block-form methods are based on voxel decompositions and generate surfaces gluing the square faces shared by voxels of different color.

Beveled-form methods are often based on volume data sampled at regular points represented as 3D pictures. Marching Cubes [LC87] is the most used representant of this family.

A block-form method cannot be applied directly from a 3D picture; a mapping from a 3D picture to a voxel decomposition is required before calling a cuberille-like reconstruction algorithm. The most common mapping is to generate voxels centered at the lattice points and whose length is equal to the grid spacing (see Figure 18 a). An alternative is considering the voxels defined by adjacent lattice points, and whose color is determined by local neighborhood (see Figure 18 b).

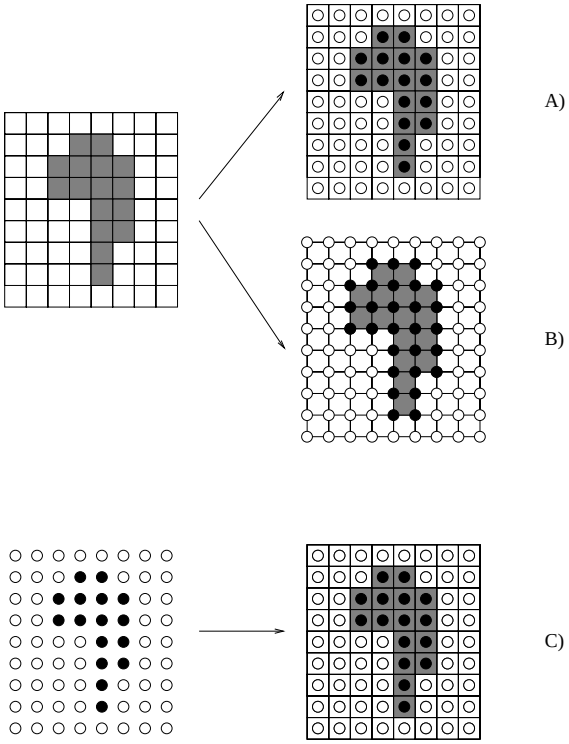


Figure 18: Mappings between voxel decompositions and pictures

Similarly, beveled-form algorithms cannot be applied directly to a voxel decomposition; a mapping to a 3D picture is required first. A simple example generates lattice points at the center of each voxel (Figure 18 c).

8.4.4 Ambiguity in surface reconstruction from 3D pictures

Since SDM only offer approximate representations of polyhedra, each SDM can be reconstructed as several slightly different polyhedra, called *interpretations*. Interpretations might be geometrically and also topologically different, yielding to the *ambiguity problem* of surface reconstruction from 3D pictures. Ambiguity occurs at several levels:

- *ambiguity at voxel edges*

  An intersected edge (i.e. an edge connecting lattice points with different color) can be interpreted as a surface intersecting the edge an odd number of times. Similarly, non-intersected edges can be interpreted as a surface intersecting the edge an even number of times. Usually, edges are interpreted as being intersected the minimum number of times, resp. one or zero (see Figure 19).

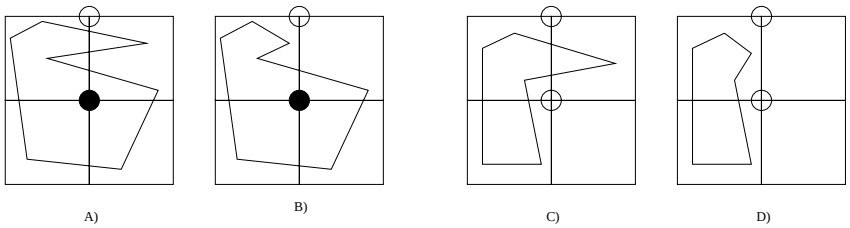


Figure 19: Ambiguity at edges: two different interpretations of intersected edges (*left*) and non-intersected edges (*right*)

- *ambiguity at voxel faces*

  An *ambiguous face* is a voxel face with a black diagonal and a white diagonal (see Figure 20). An ambiguous face has two topologically different interpretations [Dür88]: one connecting the black diagonal and the other connecting the white diagonal. Ambiguity at faces is serious because it involves surface topology and hence surface continuity. Ambiguity at faces causes the original Marching Cubes reconstruction algorithm to generate triangle meshes with some gaps. Several techniques for solving this problem have been proposed. For a review see [GW94].

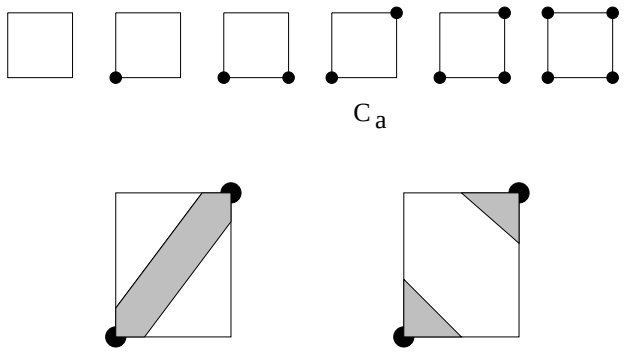


Figure 20: Voxel faces and different interpretations of ambiguous faces

- *ambiguity at voxels*

  Ambiguity at voxel level arises in a specific voxel configuration where the only two black (or white) points are placed along any of the main diagonals of the voxel (see Figure 21). Such configuration has two topologically different interpretations, with one or two connected components inside the voxel.

The most serious consequence of the ambiguity at edges is the *non-regular regions* problem, originally described in [AAB+96a]. This problem arises when sampling frequency is not high

Figure 21: An ambiguous voxel with its different interpretations

enough to capture thin surface features. Given a 3D picture, a non-regular region is a 3D volume that does not intersect any lattice point, and hence this 3D volume is not represented in the picture. Non-regular regions can be arbitrarily large (see Figure 22), although they meet the following local thickness criterion: $\forall p \in S\ \exists q \notin S \mid d_{max}(p,q) < l$, where S is the non-regular region, l is the grid spacing and $d_{max}(p,q) = max(|px - qx|, |py - qy|, |pz - qz|)$. Non-regular regions existence depends not only on grid spacing but also on grid alignment. By definition, non-regular regions intersect voxel edges an even number of times, but due to the edge ambiguity, these edges are interpreted as non-stabbed edges and hence non-regular regions are not reconstructed by isosurface extraction algorithms (see Figure 23).

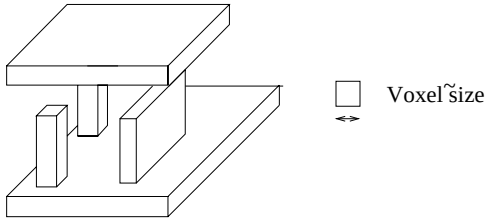


Figure 22: Example of non-regular region

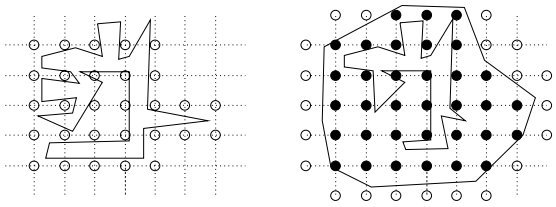


Figure 23: Non-regular region corresponding to a solid *(left)*, and to a cavity (*right*)

8.4.5 Volume buffers (He)

A mesh simplification method based on signal processing is presented in [HHK+95]. The face reduction strategy is based on discretization into a 3D picture followed by isosurface extraction. The main contribution of the method is the definition of the 3D picture from the input representation.

The two steps of the algorithm are:

- *sampling and filtering*

  The input object is sampled and low-pass filtered resulting in a 3D picture where each

lattice point (i, j, k) has assigned a density $f(i, j, k)$. The density value depends on the amount of solid in the point's neighborhood, and is computed as the following integral restricted to the solid:

$$f(i,j,k) = \int\int\int h(x,y,z)dxdydz \tag{9}$$

where $h(x, y, z)$ is a low-pass filter with an spheric support of radius R and linearly weighted so that solid contribution is maximum in the center c and null at distance R: $h(p) = (R - dist(p, c))/R$. Actually, this filter with finite support is approximated by a pre-computed discrete filter.

- *isosurface extraction*

  The result simplification is generated by applying an isosurface extraction algorithm to the 3D picture. The isosurface extraction method adopted is Marching Cubes. The user or the application must provide an isodensity value.

This method can be easily extended to support simplification of non-polygonal representations provided that they represent a closed 3D volume. Since the low-pass filter removes high-frequency details, the result representations are more smooth than the input representation. Unfortunately, Marching Cubes produces verbose representations, involving many faces even in low-curvature regions. The approximation error is not bounded.

Voxel Based Object Simplification	
INPUT	
REPRESENTATION SCHEME	Unrestricted
GEOMETRIC DOMAIN	Unrestricted
TOPOLOGY DOMAIN	Surface enclosing a 3D volume
FOCUS DOMAIN	Smooth surfaces
APPEARANCE PRESERVATION	No
APPROXIMATION ERROR	
USER-DEFINED PARAMETERS	Grid spacing and filter radius
REDUCTION PARAMETER	Grid spacing
ERROR MEASUREMENT	No
BOUNDED ERROR	No
ERROR REPORT	No
OUTPUT	
OUTPUT SCHEME	Triangle mesh
GEOMETRIC PROPERTIES	Two-manifold; no sharp edge preservation
VERTEX CREATION	Yes
TOPOLOGY PRESERVATION	No
SIMPLIFICATION LIMIT	Octahedron
USABILITY	Good
TOPOLOGY TESTS	N/A
SMOOTH TRANSITION	Anti-aliasing using several layers is reported

8.4.6 Simplification using face octrees (Joan-Arinyo et al.)

An isosurface extraction method suitable for SDM-based simplification is presented in [JAS95]. The method generates low-verbosity polygonal models from a voxel decomposition.

The four steps of the algorithm are:

- *generate quadrilateral mesh*

  The first step computes a quadrilateral mesh covering the set of boundary voxels. A voxel is said to be a boundary voxel when at least one of its 26-neighbors is white. Quadrilaterals are defined by connecting the centers of four 6-adjacent boundary voxels.

- *relax quadrilateral mesh*

  The quadrilateral mesh is relaxed using an energy function for moving mesh vertices, as in Geometrically Deformed Models (GDM for short) [MBL+91]. Relaxation is constrained so that the vertices of the GDM lie inside the voxel they represent. The relaxed GDM may include dangling quadrilaterals, which are removed from the mesh.

- *generate face octree*

  A plane is computed for each vertex of the relaxed GDM. The plane is defined so it contains the vertex and its normal vector is computed as the normal average of neighbor triangles. A face octree [Bru90] is generated by merging face nodes as much as possible using as tolerance the length of the main diagonal of voxels.

- *surface extraction*

  A surface is extracted from the face octree and the relaxed GDM. For each face node n of the face octree, the GDM vertices that lie inside n are projected into the plane associated to n. Quadrilaterals whose four projected vertices lie inside node n are merged together to generate a planar polygon. Quadrilaterals whose projected vertices belong to different face nodes are triangulated.

The main contribution of the reconstruction is that face's size is not limited by voxel's size. This method can be used for surface and solid simplification using the basic scheme of SDM-based simplification algorithms (see table below). However, the reconstruction method is unable to reconstruct non-regular regions and some regular regions, for instance regions of 2x2xn voxels. Approximation error is bounded only in reconstructed regions.

Simplification using face octrees (Joan-Arinyo et al.)	
Input	
Representation scheme	Any scheme, but must be translated to a voxel decomposition
Geometric domain	Unrestricted
Topology domain	Surface must enclose a 3D volume
Focus domain	Unrestricted
Appearance preservation	No
Approximation error	
User-defined parameters	Voxel size or number of divisions
Reduction parameter	Voxel size or number of divisions
Error measurement	Hausdorff distance
Bounded error	No
Error report	No
Output	
Output Scheme	Polyhedral model
Geometric properties	Two-manifold, no self-intersecting
Vertex creation	Yes
Topology preservation	No
Simplification limit	Empty model (no faces)
Usability	Good
Topology tests	N/A
Smooth transition	No

8.4.7 SDM-simplification using colored octrees (Andujar et al.)

An octree-based surface simplification algorithm is presented in [AAB+96a]. Given a polyhedral solid P, a multiresolution of two-manifold, polyhedral solids $s_1, \ldots, s_n$ is generated so that $distH(P, s_i) < \epsilon_i$ and $face\#(s_i) < face\#(s_{i+1})$, where $distH$ is the Hausdorff distance defined over the points on the solid's boundary and $\{\epsilon_i\}$ is an increasing series of tolerances.

The face reduction strategy is based on using an intermediate SDM called maximal division classical octree (MDCO for short) [BJN+88].

The method combines two different kinds of voxel information. On one hand, terminal nodes of the MDCO track the surface on cubic regions; on the other hand, colors at terminal nodes corners capture solid presence at punctual places.

There are $2^8 = 256$ different configurations of the eight colors of terminal nodes [LC87], [Sri81]. These configurations can be grouped by rotations ans symmetries into 14 equivalence classes. (see Figure 24).

The three steps of the algorithm are (see Figure 25):

- *BRep to MDCO conversion*

  The colored MDCO representation of the input polyhedron is generated by two simultaneous processes of recursive space subdivision and clipping of polyhedron's faces [CH88].

  TG nodes with all its vertices with the same color are called *non-regular* nodes; otherwise, they are called *regular* nodes. The following figure shows proposed classification of terminal nodes:

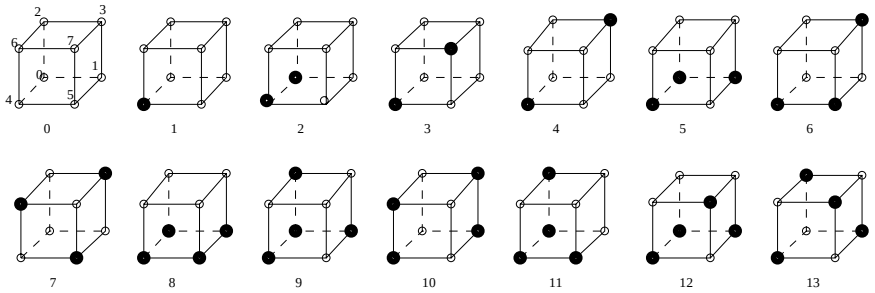
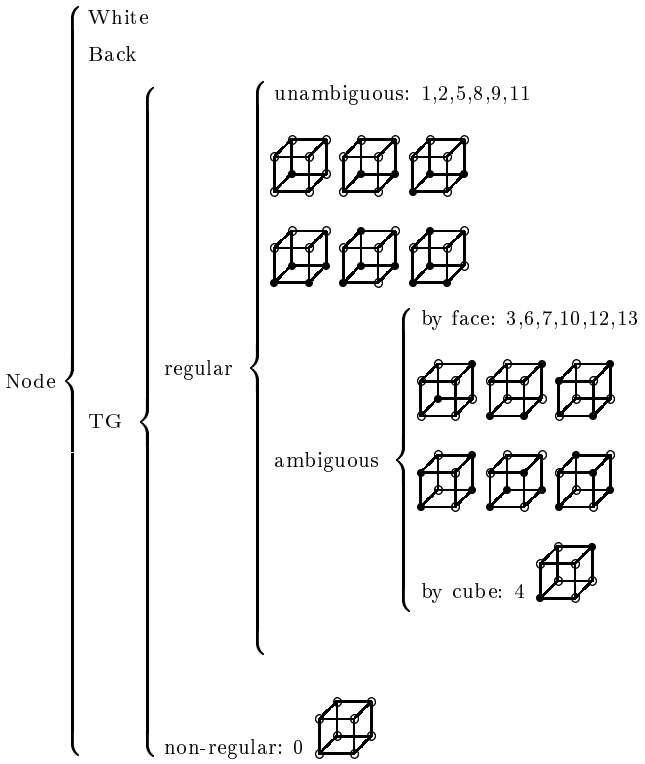


Figure 24: The 14 equivalence classes of voxels



- *generate multiresolution of MDCO*

  A multiresolution series of octrees with decreasing depth is generated by iteratively pruning the last level of the octree (see Figure 15).

- *deambiguate TG nodes*

  Ambiguous TG nodes are subdivided so that resulting nodes are unambiguous. Then, TG nodes are classified to define a discrete topology where *in* nodes belong to discrete faces, *on* nodes belong to discrete edges and *join* nodes are discrete vertices (see Figure 25). The resulting structure is called *TG-map*.

- *refine TG-map*

  The TG-map is refined until it defines a realizable topology, i.e., until there exists a feasible polyhedron with the topology defined by the TG-map. Refinement is con-

ducted by minimization of an energy function penalizing non-planar of discrete faces and non-linearly separable regions which cause non-planar faces to be splited and *on* and *join* nodes to be moved.

- *geometry realization*

  A polyhedron is generated from the TG-map. A vertex is created for each *join* node and the incidence graph is defined by *on* and *in* nodes. Geometry is generated by moving vertices minimizing the energy function.

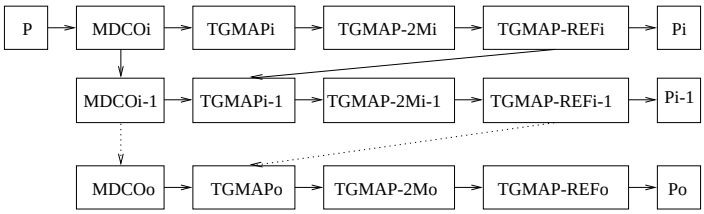


Figure 25: SDM-based simplification using colored MDCO

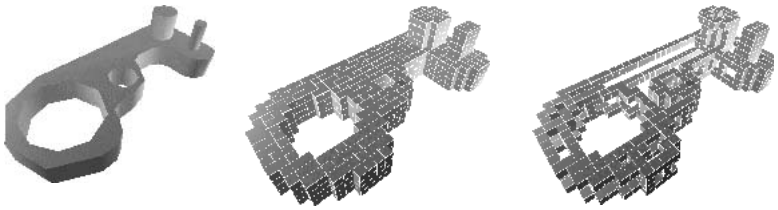


Figure 26: (a) Original solid; (b) in, on and join nodes; (c) on and join nodes

Generated polyhedra are two-manifold and the approximation error is bounded in the Hausdorff distance sense. Unlike previous approaches, this method produces faces with arbitrary size and complexity (even with holes).

SDM-simplification using colored octrees (Andujar et al.)	
Input	
Representation scheme	Polyhedral model
Geometric domain	Unrestricted
Topology domain	Surface must enclose a 3D volume
Focus domain	Unrestricted
Appearance preservation	No
Approximation error	
User-defined parameters	Octree depth
Reduction parameter	Octree depth
Error measurement	Hausdorff distance
Bounded error	Yes
Error report	Yes
Output	
Output Scheme	Polyhedral model
Geometric properties	Two-manifold, no selfintersecting
Vertex creation	Yes
Topology preservation	No
Simplification limit	Cube
Usability	Good
Topology tests	N/A
Smooth transition	No

8.4.8 Discretized Marching Cubes (Montani et al.)

A less verbose version of the original Marching Cubes, called DiscMC, is presented in [MSS94]. The surface is enforced to intersect the edges at the midpoint, instead of computing the intersection point by interpolation. Final vertices can be created only at 13 different locations of a voxel (see Figure 27a), and faces may have only 13 different plane orientations with respect to a voxel, $x = c$, $y = c$, $z = c$, $x \pm y = c$,$x \pm z = c$, $y \pm z = c$, $x \pm y \pm z = c$.

The look-up table is similar to that of MC, but some classes are reconstructed using rectangles instead of triangles. The ambiguity problem of MC is avoided by explicitly considering the complements of classes 3, 6 y 7 (see Figure 24).

The two steps of the surface reconstruction are:

- *generate discrete faces*

  For each voxel intersected by the surface, DiscMC uses the look-up table to generate the triangles and rectangles inside the voxel. Generated faces are inserted in any of 13 hash tables depending on face orientation. For indexing the hash table, shape code and cube index are used.

- *coplanar faces merging*

  Each hash table is examined for adjacent faces. Adjacent coplanar faces are merged with the help of Freeman chains.

This surface extraction technique can be used in a SDM-based simplification algorithm. Unfortunately, planar regions are reconstructed verbosily when their orientations differ from the 13 DiscMC orientations (see Figure 27b).

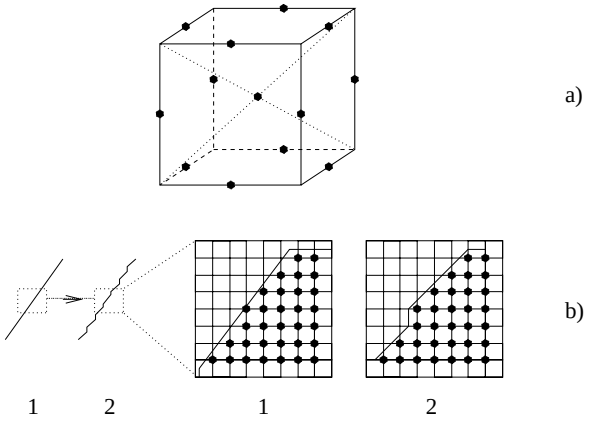


Figure 27: a) Possible vertex locations in DiscMC; b) planar regions of the original isurface (left) are simplified if their orientation coincides with any of the 13 DiscMC face orientations, and complicated otherwise (right)

Discretized Marching Cubes (Montani et al.)	
INPUT	
REPRESENTATION SCHEME	Any scheme, but must be translated to a 3D picture
GEOMETRIC DOMAIN	Unrestricted
TOPOLOGY DOMAIN	Surface must enclose a 3D volume
FOCUS DOMAIN	Volume data
APPEARANCE PRESERVATION	No
APPROXIMATION ERROR	
USER-DEFINED PARAMETERS	Voxel size of number of divisions
REDUCTION PARAMETER	Voxel size of number of divisions
ERROR MEASUREMENT	Hausdorff distance
BOUNDED ERROR	No (non-regular regions)
ERROR REPORT	No
OUTPUT	
OUTPUT SCHEME	Polyhedral model
GEOMETRIC PROPERTIES	Two-manifold; only 13 orientations for faces
VERTEX CREATION	Yes
TOPOLOGY PRESERVATION	No
SIMPLIFICATION LIMIT	Octahedron
USABILITY	Good
TOPOLOGY TESTS	N/A
SMOOTH TRANSITION	No

8.4.9 Mesh Propagation (Howie)

Another surface reconstruction method from volume data is presented in [HB94]. The method generates the same triangle mesh than a corrected MC, but represented more ef-

ficiently using triangle strips. Triangle strips allow the storage of t triangles using $t + 2$ vertices, and in addition to require less space, they can be rendered more efficiently by graphics hardware.

Like other surface extraction approaches, Mesh Propagation can be used for a SDM-based simplification method. Unfortunately, the method reduces storage requirements but does not reduce the number of faces of verbose MC meshes.

Mesh Propagation	
Input	
Representation scheme	Any scheme, converted into voxels
Geometric domain	Unrestricted
Topology domain	Unrestricted
Focus domain	Volume data
Appearance preservation	No
Approximation error	
User-defined parameters	Voxel size or number of divisions
Reduction parameter	Voxel size or number of divisions
Error measurement	Hausdorff distance
Bounded error	No
Error report	No
Output	
Output Scheme	Triangle Strips
Geometric properties	Like Marching Cubes
Vertex creation	Yes
Topology preservation	No
Simplification limit	Octahedron
Usability	Good
Topology tests	N/A.
Smooth transition	No.

9 Comparison of simplification algorithms

9.1 Comparative tables

9.1.1 Domain

All simplification methods based on SDM are restricted to surfaces enclosing a 3D volume, since the SDM construction requires the point-inside-solid query. The rest of methods are devoted to triangle meshes or triangulate the faces at the beginning (Figure 28). Triangle mesh representation is easier to simplify than other polyhedral representations because the simplicity of its incidence relations.

[SR94]	Triangle mesh from DTM
[HDD+93]	Two-manifold triangle mesh
[RB93]	Triangle mesh
[SZL92]	Triangle mesh
[Tur92]	Two-manifold triangle mesh
[ea96]	Two-manifold triangle mesh
[Ham94]	Two-manifold triangle mesh
[KT93]	Polyhedral model
[Eea95]	Surface enclosing a 3D volume
[HHK+95]	Surface enclosing a 3D volume
[JAS95]	Surface enclosing a 3D volume
[AAB+96a]	Surface enclosing a 3D volume

Figure 28: Domain

9.1.2 Support to visual information

Only a few methods take into account visual information during the simplification process (see [CPD+96], [Hop97]). Some methods provide a partial support: since the set of final vertices is a subset of the input ones, per-vertex information (such as normal or texture coordinates) can be imported from the input vertices. SDM-based methods do not provide any support to appearance preservation (Figure 29).

9.1.3 Error bounded

The absence of reasonable bounds of the approximation error is still the most important limitation of many surface simplification methods (see Figure 30).

9.1.4 Output models

The majority of surface simplification methods produce triangle meshes. Only some SDM-based methods are capable of producing faces of arbitrary complexity (Figure 31). Polyhedral representations have a number of advantages with respect to triangle meshes: they are more general (a triangle mesh is a particular case of polyhedral representation) and concise. Polyhedral representations are specially suitable in operations whose running times depend on number of faces instead of the number of vertices, such as backface culling and

[SR94]	Partial
[HDD+93]	No
[RB93]	No
[SZL92]	Partial
[Tur92]	No
[ea96]	Partial
[Ham94]	No
[KT93]	Partial
[Eea95]	No
[HHK+95]	No
[JAS95]	No
[AAB+96a]	No

Figure 29: Support to visual information

[SR94]	No
[HDD+93]	No
[RB93]	Yes
[SZL92]	No
[Tur92]	No
[ea96]	Yes
[Ham94]	No
[KT93]	Yes
[Eea95]	Yes
[HHK+95]	No
[JAS95]	No
[AAB+96a]	Yes

Figure 30: Error bounded

operations involving BSP trees. Triangulated representations and triangle strips are often preferred for real-time visualization, but triangulation of a polyhedral model is loss-less and straightforward.

9.1.5 Topology simplification

Only SDM-based methods simplify topology while maintaining a two-manifold surface. The rest of methods preserve topology, or simplify it producing invalid BReps (Figure 32).

9.2 Limitations of current simplification methods

The main limitations of current simplification methods are:

- lack of topology simplification, required for coarse representations of very complex objects,
- invalid representations, i.e. non-manifold, selfintersecting, isolated segments, etc.,
- unbounded approximation error,

[SR94]	Triangle mesh
[HDD+93]	Triangle mesh
[RB93]	Mesh of triangles, segments and points
[SZL92]	Triangle mesh
[Tur92]	Triangle mesh
[ea96]	Triangle mesh
[Ham94]	Triangle mesh
[KT93]	Triangle mesh
[Eea95]	Triangle mesh
[HHK+95]	Triangle mesh
[JAS95]	Polyhedron
[AAB+96a]	Polyhedron

Figure 31: Output models

[SR94]	No
[HDD+93]	No
[RB93]	Yes
[SZL92]	No
[Tur92]	No
[ea96]	No
[Ham94]	No
[KT93]	No
[Eea95]	No
[HHK+95]	Yes
[JAS95]	Yes
[AAB+96a]	Yes

Figure 32: Topology simplification

- limitation to triangle meshes (suitable for visualization but not as generic representation).

From the point of view of visualization applications, appearance preserving methods are still rare and current methods produce triangle meshes instead of triangle strips.

9.3 Concluding remarks

A general simplification algorithm should have these properties:

- capable of accepting arbitrary polyhedral models as input,
- capable of producing valid polyhedral models with arbitrary faces,
- possibility of selecting topology preservation or simplification,
- high scalability and usability: intuitive reduction parameters, and no human intervention required.

For visualization-oriented methods:

- appearance preservation,
- capable of producing tri-strips and quad-strips,
- capable of smooth transitions.

References

[AAB95a] D. Ayala, C. Andújar, and P. Brunet. Automatic simplification of orthogonal polyhedra. In D.W. Fellner, editor, *Modelling Virtual Worlds Distribuited Graphics*, pages 137–147. Internationallen Workshop MVD'95, Infix, 1995.

[AAB95b] D. Ayala, C. Andújar, and P. Brunet. MDCO to BRep conversion algorithm. Technical Report LSI-95-16-R, Universitat Politècnica de Catalunya. Dept. Llenguatges i Sistemes Informàtics, 1995.

[AAB+96a] C. Andújar, D. Ayala, P. Brunet, R. Joan, and J. Solé. Automatic generation of multiresolution boundary representations. *Computer Graphics Forum*, 15(3), 1996.

[AAB+96b] C. Andújar, D. Ayala, P. Brunet, R. Joan, and J. Solé. Automatic generation of multiresolution boundary representations. Technical Report LSI-96-2-R, Universitat Politècnica de Catalunya. Dept. Llenguatges i Sistemes Informàtics, 1996.

[AAB+97a] F. Alonso, C. Andújar, P. Brunet, L. García, Isabel Navazo, and A. Vinacua. Virtual reality tools in shipbuilding design, 1997. Proceedings of TeamCAD: GVU/NIST Workshop on Collaborative Design, Atlanta, Georgia.

[AAB97b] C. Andújar, D. Ayala, and P. Brunet. Volume-based polyhedra simplification using tg-maps. Technical Report LSI-97-21-R, Universitat Politècnica de Catalunya. Dept. Llenguatges i Sistemes Informàtics, 1997.

[ABAB94] C. Andújar, J. Burguera, D. Ayala, and P. Brunet. Alice: Un visualizador para entornos de realidad virtual. In *CEIG'94. IV Congreso Español de Informática Gráfica*, Zaragoza, 1994.

[And98a] Carlos Andújar. The discretized polyhedra simplification: A framework for polyhedra simplification based on decomposition schemes. Technical report, Universitat Politecnica de Catalunya, LSI-98-XR, 1998.

[And98b] Carlos Andujar. Simplificacion de modelos poliedricos (written in spanish). Technical report, Universitat Politecnica de Catalunya, LSI-98-1T, 1998.

[And98c] Carlos Andújar. Space efficient connectivity test for n-dimensional images. *Computers & Graphics*, 22(4):557–558, August 1998. ISSN 0097-8493.

[ANM97] Andrea L. Ames, David R. Nadeau, and John L. Moreland. *The VRML 2.0 sourcebook*. Wiley, New York, NY, USA, second edition, 1997.

[AS96] M.-E. Algorri and F. Schmitt. Mesh simplification. *Computer Graphics Forum*, 15(3):C77–C86, September 1996.

[Bal81] D.H. Ballard. Strip trees: a hierarchical representation for curves. *Communications ACM*, 24(5):1 – 28, 1981.

[BBCS96] Chandrajit L. Bajaj, Fausto Bernardini, Jindong Chen, and Daniel Schikore. Automatic reconstruction of 3d cad models. In *Proc. of Theory and Practice of Geometric Modelling*, 1996.

[BJN+88] P. Brunet, R. Juan, Isabel Navazo, J. Sole, and D. Tost. *Scientific Visualization. Advances and challenges*. Academic Press, 1988.

[Bru90] P. Brunet. Face octrees. involved algorithms and applications. Report LSI-90-14, Departament de Llenguatges i sistemes informàtics, Universitat Politècnica de Catalunya, 1990.

[CCMS97] A. Ciampalini, P. Cignoni, C. Montani, and R. Scopigno. Multiresolution decimation based on global error. *The Visual Computer, Springer International*, 13(5):228–246, 1997.

[CH88] H. Chen and T. Huang. A survey of construction and manipulation of octrees. *Computer Vision, Graphics and Image Processing*, 43, 1988.

[Cla76] J. H. Clark. Hierarchical geometric models for visible surface algorithms. *Communications of the ACM*, 19(10):547–554, 1976.

[Cla95] R. J. Clarke. *Digital compression of still Images and Video.* London, 1995.

[COM98] Jonathan Cohen, Marc Olano, and Dinesh Manocha. Appearance-preserving simplification. *Computer Graphics Proceedings, Annual Conference Series, 1998 (ACM SIGGRAPH '98 Proceedings)*, pages 115–122, 1998.

[CPD+96] A. Certain, J. Popovic, T. DeRose, T. Duchamp, D. Salesin, and W. Stuetzle. Interactive multiresolution surface viewing. In *Proc. SIGGRAPH*, pages 91–99, 1996.

[Cro82] F. C. Crow. A more flexible image generation environment. *Computer Graphics*, 16(3):9–18, 1982.

[Dau92] I. Daubechies. *Wavelets.* S.I.A.M, Philadelphia, 1992.

[DLW94] Tony D. DeRose, M. Lounsbery, and J. Warren. Multiresolution analysis for surfaces or arbitrary topological type. In *Proc. SIGGRAPH*, 1994. Course notes.

[Dür88] M.J. Dürst. Additional reference to marching cubes. *Computer Graphics*, 22(2), 1988.

[DZ91] Michael J. DeHaemer and Michael J. Zyda. Simplification of objects rendered by polygonal appriximations. *Computer & Graphics*, 15(2):175–184, 1991.

[ea96] Jonathan Cohen et al. Simplification Envelopes. *ACM SIGGRAPH '96 Proceedings*, pages 119–128, 1996.

[Eck97] George Eckel. *IRIS Performer Programmer's Guide.* Silicon Graphics, Inc. Document no. 007-1680-040, 1997.

[Eea95] Matthias Eck and et al. Multiresolution analysis of arbitrary meshes. In *Proc. SIGGRAPH*, pages 173–182, 1995.

[ET92] David Eu and Godfried Toussaint. On approximating polygonal curves in two and three dimensions. Technical Report SOCS 92.15, McGill University, School of Computer Science, 1992.

[Far90] Gerald Farin. *Curves and Surfaces for Computer-Aided Geometric Design.* Academic Press, New York, NY, USA, second edition, 1990.

[FLP89] Henry Fuchs, Marc Levoy, and Stephen M. Pizer. Interactive visualization of 3-D medical data. *Computer*, 22(8):46–51, August 1989.

[FS93] T.A. Funkhouser and C.H. Sequin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In *Proc. SIGGRAPH*, pages 247–254, 1993. Computer Graphics Proceedings, Annual Conference Series.

[FST92] T. A. Funkhouser, C. H. Sequin, and S. J. Teller. Management of large amounts of data in interactive building walkthroughs. In *ACM SIGGRAPH, Computer Graphics 1992 Symposium on interactive 3D graphics*, pages 11 – 20, 1992.

[FvDFH90] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics. Principles and Practice. Second Edition.* Addison-Wesley, 1990.

[GGS95] Markus H. Gross, R. Gatti, and O. Staadt. Fast multiresolution surface meshing. In *Proc. IEEE Visualization '95*, July 1995.

[GH95] Michael Garland and Paul S. Heckbert. Fast polygonal approximation of terrains and height fields. Technical report, CS Dept., Carnegie Mellon U., September 1995.

[GH97] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In *SIGGRAPH 97 Conference Proceedings*, pages 209–216. Addison Wesley, August 1997.

[Grü67] Branko Grünbaum. *Convex Polytopes.* Interscience Publishers, New York, 1967.

[GSG96] Markus H. Gross, Oliver G. Staadt, and Roger Gatti. Efficient triangular surface approximations using wavelets and quadtree data structures. *IEEE Transactins on Visualization and Computer Graphics*, 2(2), June 1996.

[Gue96] Andre Gueziec. Surface simplification inside a tolerance volume. Technical report, Yorktown Heights, NY 10598, March 1996. IBM Research Report RC 20440.

[GW94] Allen Van Gelder and Jane Wilhelms. Topological considerations in isosurface generation. *ACM Transactions on Graphics*, 13(4):337–375, 1994.

[Ham94] B. Hamann. A data reduction scheme for triangulated surfaces. *Computer Aided Geometric Design*, 11:197–214, 1994.

[HB94] C. T. Howie and E. H. Blake. The mesh propagation algorithm for isosurface construction. In *Proc. EUROGRAPHICS*, pages 65–74, 1994. Eurographics vol. 13 no. 3.

[HDD+93] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. In *Proc. SIGGRAPH*, pages 19–26, 1993. Computer Graphics Proceedings, Annual Conference Series.

[Hea96] Taosong He and et al. Controlled topology simplification. *IEEE Transactions on Visualization and Computer Graphics*, 2(2):171–184, 1996.

[HG94] P.S. Heckbert and M. Garland. Multiresolution modelling for fast rendering. *Proc. Graphics Interface 94*, pages 43–50, May 1994.

[HH93] P. Hinker and C. Hansen. Geometric optimization. In Gregory M. Nielson and Dan Bergeron, editors, *Proceedings of the Visualization '93 Conference*, pages 189–195, San Jose, CA, October 1993. IEEE Computer Society Press.

[HHK+95] T. He, L. Hong, A. Kaufman, A. Varshney, and S. Wang. Voxel based object simplification. In G.M. Nielson and D. Silver, editors, *Visualization'95*, pages 296–303, Atlanta, GA, October 29 – November 3 1995.

[HI86] Masao Iri Hiroshi Imai. Computational-geometric methods for polygonal approximations of a curve. *Computer Vision, Graphics and Image Processing*, 36:31–41, 1986.

[Hop96] H. Hoppe. Progressive meshes. *Computer Graphics*, 30(Annual Conference Series):99–108, 1996.

[Hop97] Hugues Hoppe. View-dependent refinement of progressive meshes. In *SIGGRAPH 97 Proceedings*, pages 189–198. ACM SIGGRAPH, Addison Wesley, August 1997. ISBN 0-89791-896-7.

[II88] H. Imai and M. Iri. Polygonal approximation of a curve, formulations and algorithms. In G.T. Toussaint, editor, *Computational Morphology*, pages 71 – 86. Elsevier Science Publishers B.V. (North Holland), 1988.

[JAS95] Robert Juan-Arinyo and Jaume Solé. Constructing face octrees from voxel-based volume representations. *Computer-Aided Design*, 27(10):783–791, 1995.

[Joi90] Joint Photographic Experts Group ISO/IEC, JTC/SC/WG8, CCITT SGVIII. JPEG technical specifications, revision 5. *Report JPEG-8-R5*, January 1990.

[Kal92] Alan Kalvin. A survey of algorithms for constructing surfaces from 3d volume data. Technical Report RC 17600, IBM Research Division. T.J. Watson Research Center., 1992.

[KCHN91] Alan D. Kalvin, Court B. Cutting, B. Haddad, and M. E. Noz. Constructing topologically connected surfaces for the comprehensive analysis of 3D medical structures. In *Medical Imaging V: Image Processing*, volume 1445, pages 247–258. SPIE, February 1991.

[KR89] T. Y. Kong and A. Rosenfeld. Digital topology: Introduction and survey. *Computer Vision, Graphics and Image Processing*, 48:357–393, 1989.

[KS96] Reinhard Klein and W. Straber. Generation of multiresolution models from cad data for real time rendering. In *Proc. of Theory and Practice of Geometric Modelling*, 1996.

[KT93] Alan Kalvin and Russell Taylor. Superfaces: Polyhedral approximation with bounded error. Technical Report RC19135, IBM Research Division. T.J. Watson Research Center., 1993.

[KT96] Alan D. Kalvin and Russell H. Taylor. Superfaces: Polygonal mesh simplification with bounded error. *IEEE Computer Graphics and Applications*, ?(?):64–77, 1996.

[LC87] William Lorensen and Harvey Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proc. SIGGRAPH*, pages 44–50, 1987. Computer Graphics vol. 21 no. 4.

[Man88] Martti Mantyla. *An Introduction to Solid Modeling.* Computer Science Press, Rockville, MD, 1988.

[MBL+91] James Miller, David Breen, Willian Lorensen, Robert OBara, and Michael Wozny. Geometrically deformed models: A method for extracting closed geometric models from volume data. In *Proc. SIGGRAPH*, pages 217–226, 1991. Computer Graphics, vol. 25 no. 4.

[MPFL96] Joan L. Mitchell, William B. Pennebaker, Chad E. Frogg, and Didier J. Legall. *MPEG Video Compression Standard.* Chapman and Hall, New York, 1996.

[MS95] Paulo W. C. Maciel and Peter Shirley. Visual navigation of large environments using textured clusters. In Pat Hanrahan and Jim Winget, editors, *1995 Symposium on Interactive 3D Graphics*, pages 95–102. ACM SIGGRAPH, April 1995. ISBN 0-89791-736-7.

[MSS94] C. Montani, R. Scateni, and R. Scopigno. Discretized marching cubes. In *Visualization'94*, pages 281–287. IEEE Computer Society Press, 1994.

[NAM96] David R. Nadeau, Andrea L. Ames, and John L. Moreland. Optimizing the performance of VRML worlds. *Dr. Dobb's Journal of Software Tools*, 21(7):16–18, 20, 22, 24, July 1996.

[OL98] Marc Olano and Anselmo Lastra. A shading language on graphics hardware: The pixelflow shading system. *Computer Graphics Proceedings, Annual Conference Series, 1998 (ACM SIGGRAPH '98 Proceedings)*, pages 159–168, 1998.

[PH97] Jovan Popović and Hugues Hoppe. Progressive simplicial complexes. In *SIGGRAPH 97 Conference Proceedings*, pages 217–224. ACM SIGGRAPH, Addison Wesley, August 1997. ISBN 0-89791-896-7.

[PM85] F. Preparata and M.Shamos. *Computational Geometry.* Springer Verlag, 1985.

[PRS89] A. Paoluzzi, M. Ramella, and A. Santarelli. Boolean algebra over linear polyhedra. *Computer-Aided Design*, 21(8):474–484, 1989.

[PTN93] Anna Puig, D. Tost, and Isabel Navazo. Estat de l'art en modelatge i visualitzacio de volum. *Report LSI-93-5-T*, 1993.

[RB93] J. Rossignac and P. Borrel. Multiresolution 3D aproximations for rendering complex scenes. In *Modeling in Computer Graphics.* Springer-Verlag, 1993.

[RBC+92] Mary Beth Ruskai, Gregory Beylkin, Ronald Coifman, Ingrid Daubechies, Stephane Mallat, Yves Meyer, and Louise Raphael, editors. *Wavelets and Their Applications.* Jones and Bartlett Publishers, 1992.

[Red96] M. Reddy. SCROOGE: Perceptually-driven polygon reduction. *Computer Graphics Forum*, 15(4):191–203, 1996. ISSN 0167-7055.

[Req80] Aristides A. G. Requicha. Representations for rigid solids:theory, methods and systems. *ACM Computing Surveys*, 12(4):437–463, 1980.

[RH94] John Rohlf and James Helman. IRIS performer: A high performance multiprocessing toolkit for real–Time 3D graphics. In Andrew Glassner, editor, *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pages 381–395. ACM SIGGRAPH, ACM Press, July 1994. ISBN 0-89791-667-0.

[RR96] R. Ronfard and J. Rossignac. Full-range approximation of triangulated polyhedra. *Computer Graphics Forum*, 15(3):C67–C76, C462, September 1996.

[Sak95] Susumu Sakakibara. *A Beginner's Guide to Wavelets.* Tokyo Denki Daigaku, Tokyo, Japan, 1995.

[Sam90a] H. Samet. Applications of spatial data structures. *Computer Graphics, Image Processing and GIS*, 1990.

[Sam90b] H. Samet. *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS.* Addison-Wesley, Reading, MA, 1990.

[Sam90c] Hanan Samet. *The design and analysis of spatial data structures.* Reading, Mass. : Addison-Wesley, 1990, 493 p., (Addison-Wesley series in computer science), 1990, 1990.

[SL96] Marc Soucy and Denis Laurendeau. Multiresolution surface modeling based on hierarchical triangulation. *Computer Vision and Image Understanding: CVIU*, 63(1):1–14, January 1996.

[Smi94] Brian Smits. *Efficient Hierarchical Radiosity for Complex Environments.* Ph.D. thesis, Cornell University, Ithaca, NY, 1994.

[SR94] Florian Schroder and Patrick Robbach. Managing the complexity of digital terrain models. *Computer & Graphics*, 18(6):775–783, 1994.

[Sri81] Sargur N. Srihari. Representation of three-dimensional digital images. *Computing Surveys*, 13(4):399–424, 1981.

[SZL92] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen. Decimation of triangle meshes. In *Proc. SIGGRAPH*, pages 65–70, 1992. Computer Graphics vol. 26 no. 2.

[Tan97] Kok-Lim Low Tiow-Seng Tan. Model simplification using vertex-clustering (color plate S. 188). In *Proceedings of the Symposium on Interactive 3D Graphics*, pages 75–82, New York, April27–30 1997. ACM Press.

[TH93] Seth Teller and Pat Hanrahan. Global visibility algorithms for illumination computations. In *Proc. SIGGRAPH*, pages 239–246, 1993. Computer Graphics.

[Tur92] Greg Turk. Re-tiling polygonal surfaces. In *Proc. SIGGRAPH*, pages 55–64, 1992. Computer Graphics vol. 26 no. 2.

[Vel92] R.C. Veltkamp. *Closed Object Boundaries from Scattered Points.* PhD thesis, Erasmus University Rotterdam, 1992.

[Vel93] Remco C. Veltkamp. 3D computational morphology. In R. J. Hubbold and R. Juan, editors, *Eurographics '93*, pages 115–127, Oxford, UK, 1993. Eurographics, Blackwell Publishers.

[YL95] B-L. Yeo and B. Liu. A unified approach to temporal segmentation of motion JPEG and MPEG compressed video. In *International Conference on Multimedia Computing and Systems.* IEEE Computer Society, May 1995.

[You94] R. K. Young. *Wavelet Theory and Its Applications.* Kluwer Academic Pub., 1994.

[AAB+96a] [AAB95a] [AAB+97a] [ABAB94] [AAB95b] [AAB97b] [AAB+96b] [And98c] [And98a]