Progressive Simplicial Complexes

Jovan Popović*
Carnegie Mellon University

Hugues Hoppe
Microsoft Research

ABSTRACT

In this paper, we introduce the progressive simplicial complex (PSC) representation, a new format for storing and transmitting triangulated geometric models. Like the earlier progressive mesh (PM) representation, it captures a given model as a coarse base model together with a sequence of refinement transformations that progressively recover detail. The PSC representation makes use of a more general refinement transformation, allowing the given model to be an arbitrary triangulation (e.g. any dimension, non-orientable, non-manifold, non-regular), and the base model to always consist of a single vertex. Indeed, the sequence of refinement transformations encodes both the geometry and the topology of the model in a unified multiresolution framework. The PSC representation retains the advantages of PM's. It defines a continuous sequence of approximating models for runtime level-of-detail control, allows smooth transitions between any pair of models in the sequence, supports progressive transmission, and offers a space-efficient representation. Moreover, by allowing changes to topology, the PSC sequence of approximations achieves better fidelity than the corresponding PM sequence.

We develop an optimization algorithm for constructing PSC representations for graphics surface models, and demonstrate the framework on models that are both geometrically and topologically complex.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling - surfaces and object representations.

Additional Keywords: model simplification, level-of-detail representations, multiresolution, progressive transmission, geometry compression.

1 INTRODUCTION

Modeling and 3D scanning systems commonly give rise to triangle meshes of high complexity. Such meshes are notoriously difficult to render, store, and transmit. One approach to speed up rendering is to replace a complex mesh by a set of level-of-detail (LOD) approximations; a detailed mesh is used when the object is close to the viewer, and coarser approximations are substituted as the object recedes [6, 8]. These LOD approximations can be precomputed automatically using mesh simplification methods (e.g. [2, 10, 14, 20, 21, 22, 24, 27]). For efficient storage and transmission, mesh compression schemes [7, 26] have also been developed.

The recently introduced *progressive mesh* (PM) representation [13] provides a unified solution to these problems. In PM form, an arbitrary mesh $\hat{M}$ is stored as a coarse base mesh M^0 together with a sequence of n detail records that indicate how to incrementally refine M^0 into $M^n = \hat{M}$ (see Figure 7). Each detail record encodes the information associated with a *vertex split*, an elementary transformation that adds one vertex to the mesh. In addition to defining a continuous sequence of approximations $M^0 \ldots M^n$, the PM representation supports smooth visual transitions (geomorphs), allows progressive transmission, and makes an effective mesh compression scheme.

The PM representation has two restrictions, however. First, it can only represent *meshes*: triangulations that correspond to orientable[1] 2-dimensional manifolds. Triangulated[2] models that cannot be represented include 1-d manifolds (open and closed curves), higher dimensional polyhedra (e.g. triangulated volumes), non-orientable surfaces (e.g. Möbius strips), non-manifolds (e.g. two cubes joined along an edge), and non-regular models (i.e. models of mixed dimensionality). Second, the expressiveness of the PM vertex split transformations constrains all meshes $M^0 \ldots M^n$ to have the same topological type. Therefore, when $\hat{M}$ is topologically complex, the simplified base mesh M^0 may still have numerous triangles (Figure 7).

In contrast, a number of existing simplification methods allow topological changes as the model is simplified (Section 6). Our work is inspired by vertex unification schemes [21, 22], which merge vertices of the model based on geometric proximity, thereby allowing genus modification and component merging.

In this paper, we introduce the *progressive simplicial complex* (PSC) representation, a generalization of the PM representation that permits topological changes. The key element of our approach is the introduction of a more general refinement transformation, the *generalized vertex split*, that encodes changes to both the geometry and topology of the model. The PSC representation expresses an arbitrary triangulated model M (e.g. any dimension, non-orientable, non-manifold, non-regular) as the result of successive refinements applied to a base model M^1 that always consists of a single vertex (Figure 8). Thus both geometric and topological complexity are recovered progressively. Moreover, the PSC representation retains the advantages of PM's, including continuous LOD, geomorphs, progressive transmission, and model compression.

In addition, we develop an optimization algorithm for constructing a PSC representation from a given model, as described in Section 4.

* Work performed while at Microsoft Research.
Email: jovan@cs.cmu.edu, hhoppe@microsoft.com
Web: http://www.cs.cmu.edu/∼jovan/
Web: http://research.microsoft.com/∼hoppe/

[1] The particular parametrization of vertex splits in [13] assumes that mesh triangles are consistently oriented.

[2] Throughout this paper, we use the words "triangulated" and "triangulation" in the general dimension-independent sense.

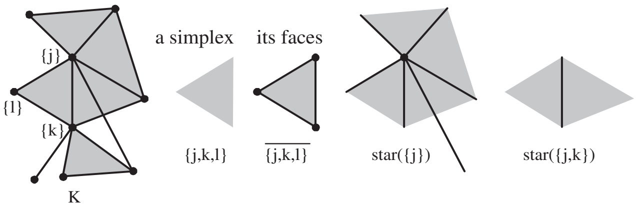


Figure 1: Illustration of a simplicial complex K and some of its subsets.

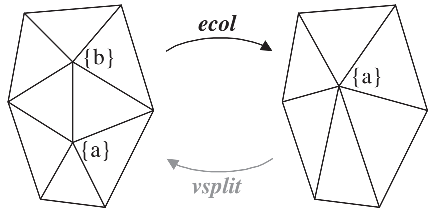


Figure 2: Illustration of the edge collapse transformation and its inverse, the vertex split.

2 BACKGROUND

2.1 Concepts from algebraic topology

To precisely define both triangulated models and their PSC representations, we find it useful to introduce some elegant abstractions from algebraic topology (e.g. [15, 25]).

The geometry of a triangulated model is denoted as a tuple (K, V) where the *abstract simplicial complex* K is a combinatorial structure specifying the adjacency of vertices, edges, triangles, etc., and V is a set of vertex positions specifying the shape of the model in $\mathbf{R}^3$.

More precisely, an abstract simplicial complex K consists of a set of vertices $\{1, \dots, m\}$ together with a set of non-empty subsets of the vertices, called the *simplices* of K, such that any set consisting of exactly one vertex is a simplex in K, and every non-empty subset of a simplex in K is also a simplex in K.

A simplex containing exactly d+1 vertices has *dimension d* and is called a d-simplex. As illustrated pictorially in Figure 1, the *faces* of a simplex s, denoted $\overline{s}$, is the set of non-empty subsets of s. The *star* of s, denoted star(s), is the set of simplices of which s is a face. The *children* of a d-simplex s are the $(d-1)$-simplices of $\overline{s}$, and its *parents* are the $(d+1)$-simplices of star(s). A simplex with exactly one parent is said to be a *boundary simplex*, and one with no parents a *principal simplex*. The dimension of K is the maximum dimension of its simplices; K is said to be *regular* if all its principal simplices have the same dimension.

To form a triangulation from K, identify its vertices $\{1, \dots, m\}$ with the standard basis vectors $\{\mathbf{e}_1, \dots, \mathbf{e}_m\}$ of $\mathbf{R}^m$. For each simplex s, let the *open simplex* $\langle s\rangle \subset \mathbf{R}^m$ denote the interior of the convex hull of its vertices:

$$\langle s\rangle = \{\mathbf{b} \in \mathbf{R}^m \ : \ \mathbf{b}_j \geq 0\ ,\ \sum_{j=1}^{m} \mathbf{b}_j = 1\ ,\ \mathbf{b}_j > 0 \Leftrightarrow \{j\} \subseteq s\}.$$

The *topological realization* $|K|$ is defined as $|K| = \langle K\rangle = \cup_{s\in K}\langle s\rangle$. The *geometric realization* of K is the image $\phi_V(|K|)$ where ϕ_V : $\mathbf{R}^m \to \mathbf{R}^3$ is the linear map that sends the j-th standard basis vector $\mathbf{e}_j \in \mathbf{R}^m$ to $\mathbf{v}_j \in \mathbf{R}^3$. Only a restricted set of vertex positions $V = \{\mathbf{v}_1, \dots, \mathbf{v}_m\}$ lead to an embedding of $\phi_V(|K|) \subset \mathbf{R}^3$, that is, prevent self-intersections. The geometric realization $\phi_V(|K|)$ is often called a *simplicial complex* or *polyhedron*; it is formed by an arbitrary union of points, segments, triangles, tetrahedra, etc. Note that there generally exist many triangulations (K, V) for a given polyhedron. (Some of the vertices V may lie in the polyhedron's interior.)

Two sets are said to be *homeomorphic* (denoted $\cong$) if there exists a continuous one-to-one mapping between them. Equivalently, they are said to have the same *topological type*. The topological realization $|K|$ is a *d-dimensional manifold without boundary* if for each vertex $\{j\}$, $\langle\text{star}(\{j\})\rangle \cong \mathbf{R}^d$. It is a *d-dimensional manifold* if each $\langle\text{star}(\{v\})\rangle$ is homeomorphic to either $\mathbf{R}^d$ or $\mathbf{R}^d_+$, where $\mathbf{R}^d_+ = \{\mathbf{x} \in \mathbf{R}^d : \mathbf{x}_1 \geq 0\}$. Two simplices s_1 and s_2 are *d-adjacent* if they have a common d-dimensional face. Two d-adjacent $(d+1)$-simplices s_1 and s_2 are *manifold-adjacent* if $\langle\text{star}(s_1 \cap s_2)\rangle \cong \mathbf{R}^{d+1}$.

Transitive closure of 0-adjacency partitions K into *connected components*. Similarly, transitive closure of manifold-adjacency partitions K into *manifold components*.

2.2 Review of progressive meshes

In the PM representation [13], a mesh with appearance attributes is represented as a tuple $M = (K, V, D, S)$, where the abstract simplicial complex K is restricted to define an orientable 2-dimensional manifold, the vertex positions $V = \{\mathbf{v}_1, \dots, \mathbf{v}_m\}$ determine its geometric realization $\phi_V(|K|)$ in $\mathbf{R}^3$, D is the set of discrete material attributes d_f associated with 2-simplices $f \in K$, and S is the set of scalar attributes $s_{(v,f)}$ (e.g. normals, texture coordinates) associated with corners (vertex-face tuples) of K.

An initial mesh $\hat{M} = M^n$ is simplified into a coarser base mesh M^0 by applying a sequence of n successive edge collapse transformations:

$$(\hat{M} = M^n) \xrightarrow{ecol_{n-1}} \dots \xrightarrow{ecol_1} M^1 \xrightarrow{ecol_0} M^0 \ .$$

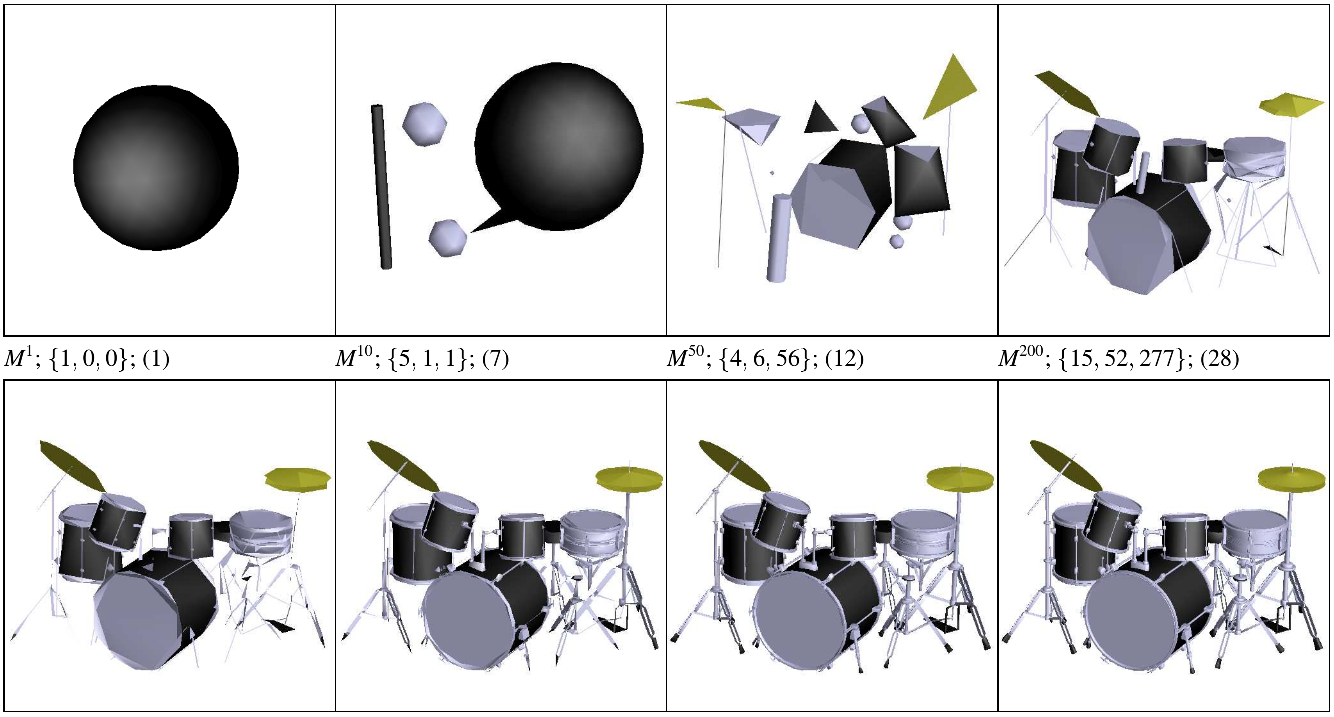
As shown in Figure 2, each *ecol* unifies the two vertices of an edge $\{a, b\}$, thereby removing one or two triangles. The position of the resulting unified vertex can be arbitrary. Because the edge collapse transformation has an inverse, called the *vertex split* transformation (Figure 2), the process can be reversed, so that an arbitrary mesh $\hat{M}$ may be represented as a simple mesh M^0 together with a sequence of n *vsplit* records:

$$M^0 \xrightarrow{vsplit_0} M^1 \xrightarrow{vsplit_1} \dots \xrightarrow{vsplit_{n-1}} (M^n = \hat{M})$$

The tuple $(M^0, \{vsplit_0, \dots, vsplit_{n-1}\})$ forms a *progressive mesh* (PM) representation of $\hat{M}$.

The PM representation thus captures a continuous sequence of approximations $M^0 \dots M^n$ that can be quickly traversed for interactive level-of-detail control. Moreover, there exists a correspondence between the vertices of any two meshes M^c and M^f $(0 \leq c < f \leq n)$ within this sequence, allowing for the construction of smooth visual transitions (geomorphs) between them. A sequence of such geomorphs can be precomputed for smooth runtime LOD. In addition, PM's support progressive transmission, since the base mesh M^0 can be quickly transmitted first, followed the *vsplit* sequence. Finally, the *vsplit* records can be encoded concisely, making the PM representation an effective scheme for mesh compression.

Topological constraints Because the definitions of *ecol* and *vsplit* are such that they preserve the topological type of the mesh (i.e. all $|K^i|$ are homeomorphic), there is a constraint on the minimum complexity that K^0 may achieve. For instance, it is known that the minimal number of vertices for a closed genus g mesh (orientable 2-manifold) is $\lceil(7+(48g+1)^{\frac{1}{2}})/2\rceil$ if $g \neq 2$ (10 if $g = 2$) [16]. Also, the presence of boundary components may further constrain the complexity of K^0. Most importantly, $\hat{K}$ may consist of a number of components, and each is required to appear in the base mesh. For example, the meshes in Figure 7 each have 117 components. As evident from the figure, the geometry of PM meshes may deteriorate severely as they approach topological lower bound.



M^1; $\{1,0,0\}$; (1) M^{10}; $\{5,1,1\}$; (7) M^{50}; $\{4,6,56\}$; (12) M^{200}; $\{15,52,277\}$; (28)

M^{500}; $\{39,68,690\}$; (58) M^{2000}; $\{14,25,3219\}$; (108) M^{5000}; $\{0,2,9010\}$; (176) $M^{n=34794}$; $\{0,0,68776\}$; (207)

Figure 3: Example of a PSC representation. The image captions indicate the number of principal $\{0,1,2\}$-simplices respectively and the number of connected components (in parenthesis).

3 PSC REPRESENTATION

3.1 Triangulated models

The first step towards generalizing PM's is to let the PSC representation encode more general triangulated models, instead of just meshes.

We denote a triangulated model as a tuple $M = (K, V, D, A)$. The abstract simplicial complex K is not restricted to 2-manifolds, but may in fact be arbitrary. To represent K in memory, we encode the *incidence graph* of the simplices using the following linked structures (in C++ notation):

```
struct Simplex {
    int dim;      // 0=vertex, 1=edge, 2=triangle, ...
    int id;
    Simplex* children[MAXDIM+1];      // [0..dim]
    List<Simplex*> parents;
};
```

To render the model, we draw only the principal simplices of K, denoted $\mathcal{P}(K)$ (i.e. vertices not adjacent to edges, edges not adjacent to triangles, etc.). The discrete attributes D associate a material identifier d_s with each simplex $s \in \mathcal{P}(K)$. For the sake of simplicity, we avoid explicitly storing surface normals at "corners" (using a set S) as done in [13]. Instead we let the material identifier d_s contain a *smoothing group* field [28], and let a normal discontinuity (*crease*) form between any pair of adjacent triangles with different smoothing groups.

Previous vertex unification schemes [21, 22] render principal simplices of dimension 0 and 1 (denoted $\mathcal{P}_{01}(K)$) as points and lines respectively with fixed, device-dependent screen widths. To better approximate the model, we instead define a set A that associates an area $a_s \in A$ with each simplex $s \in \mathcal{P}_{01}(K)$. We think of a 0-simplex $s_0 \in \mathcal{P}_0(K)$ as approximating a sphere with area a_{s_0}, and a 1-simplex $s_1 = \{j, k\} \in \mathcal{P}_1(K)$ as approximating a cylinder (with axis $(\mathbf{v}_j, \mathbf{v}_k)$) of area a_{s_1}. To render a simplex $s \in \mathcal{P}_{01}(K)$, we determine the radius r_{model} of the corresponding sphere or cylinder in modeling space, and project the length r_{model} to obtain the radius r_{screen} in screen pixels. Depending on r_{screen}, we render the simplex as a polygonal sphere or cylinder with radius r_{model}, a 2D point or line with thickness $2r_{screen}$, or do not render it at all. This choice based on r_{screen} can be adjusted to mitigate the overhead of introducing polygonal representations of spheres and cylinders.

As an example, Figure 3 shows an initial model $\hat{M}$ of 68,776 triangles. One of its approximations M^{500} is a triangulated model with $\{39, 68, 690\}$ principal $\{0,1,2\}$-simplices respectively.

3.2 Level-of-detail sequence

As in progressive meshes, from a given triangulated model $\hat{M} = M^n$, we define a sequence of approximations M^i:

$$M^1 \overset{op_1}{\longleftrightarrow} M^2 \overset{op_2}{\longleftrightarrow} \ldots \; M^{n-1} \overset{op_{n-1}}{\longleftrightarrow} M^n \; .$$

Here each model M^i has exactly i vertices. The simplification operator $M^i \overset{vunify_i}{\longleftarrow} M^{i+1}$ is the *vertex unification* transformation, which merges two vertices (Section 3.3), and its inverse $M^i \overset{gvspl_i}{\longrightarrow} M^{i+1}$ is the *generalized vertex split* transformation (Section 3.4). The tuple $(M^1, \{gvspl_1, \ldots, gvspl_{n-1}\})$ forms a *progressive simplicial complex* (PSC) representation of $\hat{M}$.

To construct a PSC representation, we first determine a sequence of *vunify* transformations simplifying $\hat{M}$ down to a single vertex, as described in Section 4. After reversing these transformations, we renumber the simplices in the order that they are created, so that each $gvspl_i(\{a_i\}, \ldots)$ splits the vertex $\{a_i\} \in K^i$ into two vertices $\{a_i\}, \{i{+}1\} \in K^{i+1}$. As vertices may have different positions in the different models, we denote the position of $\{j\}$ in M^i as $\mathbf{v}^i_j$.

To better approximate a surface model $\hat{M}$ at lower complexity levels, we initially associate with each (principal) 2-simplex s an area a_s equal to its triangle area in $\hat{M}$. Then, as the model is simplified, we keep constant the sum of areas a_s associated with principal simplices within each manifold component. When 2-simplices are eventually reduced to principal 1-simplices and 0-simplices, their associated areas will provide good estimates of the original component areas.

3.3 Vertex unification transformation

The transformation $\mathit{vunify}(\{a_i\},\{b_i\},\mathit{midp}_i) : M^i \leftarrow M^{i+1}$ takes an arbitrary pair of vertices $\{a_i\},\{b_i\} \in K^{i+1}$ (simplex $\{a_i, b_i\}$ need not be present in K^{i+1}) and merges them into a single vertex $\{a_i\} \in K^i$.

Model M^i is created from M^{i+1} by updating each member of the tuple (K, V, D, A) as follows:

K**:** References to $\{b_i\}$ in all simplices of K are replaced by references to $\{a_i\}$. More precisely, each simplex s in $\mathrm{star}(\{b_i\}) \subset K^{i+1}$ is replaced by simplex $(s \setminus \{b_i\}) \cup \{a_i\}$, which we call the *ancestor simplex* of s. If this ancestor simplex already exists, s is deleted.

V**:** Vertex $\mathbf{v}_b$ is deleted. For simplicity, the position of the remaining (unified) vertex is set to either the midpoint or is left unchanged. That is, $\mathbf{v}^i_a = (\mathbf{v}^{i+1}_a + \mathbf{v}^{i+1}_b)/2$ if the boolean parameter midp_i is *true*, or $\mathbf{v}^i_a = \mathbf{v}^{i+1}_a$ otherwise.

D**:** Materials are carried through as expected. So, if after the vertex unification an ancestor simplex $(s \setminus \{b_i\}) \cup \{a_i\} \in K^i$ is a new principal simplex, it receives its material from $s \in K^{i+1}$ if s is a principal simplex, or else from the single parent $s \cup \{a_i\} \in K^{i+1}$ of s.

A**:** To maintain the initial areas of manifold components, the areas a_s of deleted principal simplices are redistributed to manifold-adjacent neighbors. More concretely, the area of each principal d-simplex s deleted during the K update is distributed to a manifold-adjacent d-simplex not in $\mathrm{star}(\{a_i, b_i\})$. If no such neighbor exists and the ancestor of s is a principal simplex, the area a_s is distributed to that ancestor simplex. Otherwise, the manifold component ($\mathrm{star}(\{a_i, b_i\})$) of s is being squashed between two other manifold components, and a_s is discarded.

3.4 Generalized vertex split transformation

Constructing the PSC representation involves recording the information necessary to perform the inverse of each vunify_i. This inverse is the generalized vertex split gvspl_i, which splits a 0-simplex $\{a_i\}$ to introduce an additional 0-simplex $\{b_i\}$. (As mentioned previously, renumbering of simplices implies $b_i \equiv i{+}1$, so index b_i need not be stored explicitly.) Each gvspl_i record has the form

$$\mathit{gvspl}_i(\{a_i\}, C^{\Delta K}_i, \mathit{midp}_i, (\Delta\mathbf{v})_i, C^{\Delta D}_i, C^{\Delta A}_i) ,$$

and constructs model M^{i+1} from M^i by updating the tuple (K, V, D, A) as follows:

K**:** As illustrated in Figure 4, any simplex adjacent to $\{a_i\}$ in K^i can be the *vunify* result of one of four configurations in K^{i+1}. To construct K^{i+1}, we therefore replace each ancestor simplex $s \in \mathrm{star}(\{a_i\})$ in K^i by either **(1)** s, **(2)** $(s \setminus \{a_i\}) \cup \{i{+}1\}$, **(3)** s and $(s \setminus \{a_i\}) \cup \{i{+}1\}$, or **(4)** s, $(s \setminus \{a_i\}) \cup \{i{+}1\}$ and $s \cup \{i{+}1\}$. The choice is determined by a *split code* associated with s. These split codes are stored as a code string $C^{\Delta K}_i$, in which the simplices $\mathrm{star}(\{a_i\})$ are sorted first in order of increasing dimension, and then in order of increasing simplex id, as shown in Figure 5.

V**:** The new vertex is assigned position $\mathbf{v}^{i+1}_{i+1} = \mathbf{v}^i_{a_i} + (\Delta\mathbf{v})_i$. The other vertex is given position $\mathbf{v}^{i+1}_{a_i} = \mathbf{v}^i_{a_i} - (\Delta\mathbf{v})_i$ if the boolean parameter midp_i is *true*; otherwise its position remains unchanged.

D**:** The string $C^{\Delta D}_i$ is used to assign materials d_s for each new principal simplex. Simplices in $C^{\Delta D}_i$, as well as in $C^{\Delta A}_i$ below, are sorted by simplex dimension and simplex id as in $C^{\Delta K}_i$.

A**:** During reconstruction, we are only interested in the areas a_s for $s \in \mathcal{P}_{01}(K)$. The string $C^{\Delta A}_i$ tracks changes in these areas.

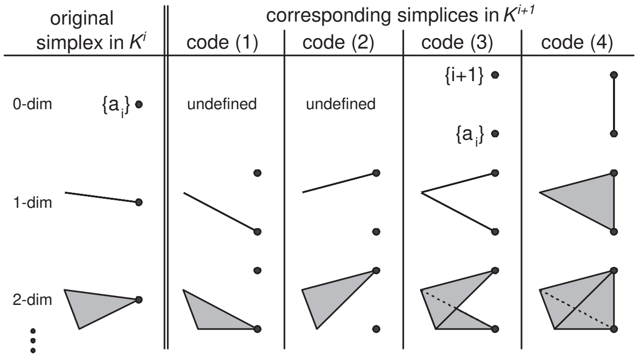


Figure 4: Effects of split codes on simplices of various dimensions.

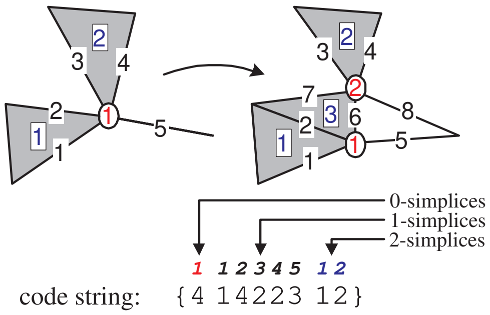


Figure 5: Example of split code encoding.

3.5 Properties

Levels of detail A graphics application can efficiently transition between models $M^1 \ldots M^n$ at runtime by performing a sequence of *vunify* or *gvspl* transformations. Our current research prototype was not designed for efficiency; it attains simplification rates of about 6000 *vunify*/sec and refinement rates of about 5000 *gvspl*/sec. We expect that a careful redesign using more efficient data structures would significantly improve these rates.

Geomorphs As in the PM representation, there exists a correspondence between the vertices of the models $M^1 \ldots M^n$. Given a coarser model M^c and a finer model M^f, $1 \le c < f \le n$, each vertex $\{j\} \in K^f$ corresponds to a unique ancestor vertex $\{\rho^{f\to c}(j)\} \in K^c$ found by recursively traversing the ancestor simplex relations:

$$\rho^{f\to c}(j) = \begin{cases} j & , j \le c \\ \rho^{f\to c}(a_{j-1}) & , j > c \end{cases} .$$

This correspondence allows the creation of a smooth visual transition (geomorph) $M^G(\alpha)$ such that $M^G(1)$ equals M^f and $M^G(0)$ looks identical to M^c. The geomorph is defined as the model

$$M^G(\alpha) = (K^f, V^G(\alpha), D^f, A^G(\alpha))$$

in which each vertex position is interpolated between its original position in V^f and the position of its ancestor in V^c:

$$\mathbf{v}^G_j(\alpha) = (\alpha)\mathbf{v}^f_j + (1-\alpha)\mathbf{v}^c_{\rho^{f\to c}(j)} .$$

However, we must account for the special rendering of principal simplices of dimension 0 and 1 (Section 3.1). For each simplex $s \in \mathcal{P}_{01}(K^f)$, we interpolate its area using

$$a^G_s(\alpha) = (\alpha)a^f_s + (1-\alpha)a^c_s ,$$

where $a^c_s = 0$ if $s \notin \mathcal{P}_{01}(K^c)$. In addition, we render each simplex $s \in \mathcal{P}_{01}(K^c) \setminus \mathcal{P}_{01}(K^f)$ using area $a^G_s(\alpha) = (1-\alpha)a^c_s$. The resulting geomorph is visually smooth even as principal simplices are introduced, removed, or change dimension. The accompanying video demonstrates a sequence of such geomorphs.

Progressive transmission As with PM's, the PSC representation can be progressively transmitted by first sending M^1, followed by the *gvspl* records. Unlike the base mesh of the PM, M^1 always consists of a single vertex, and can therefore be sent in a fixed-size record. The rendering of lower-dimensional simplices as spheres and cylinders helps to quickly convey the overall shape of the model in the early stages of transmission.

Model compression Although PSC *gvspl* are more general than PM *vsplit* transformations, they offer a surprisingly concise representation of $\hat{M}$. Table 1 lists the average number of bits required to encode each field of the *gvspl* records.

Using arithmetic coding [30], the vertex id field $\{a_i\}$ requires $\log_2 i$ bits, and the boolean parameter $midp_i$ requires 0.6–0.9 bits for our models. The $(\Delta\mathbf{v})_i$ delta vector is quantized to 16 bits per coordinate (48 bits per $\Delta\mathbf{v}$), and stored as a variable-length field [7, 13], requiring about 31 bits on average.

At first glance, each split code in the code string $C_i^{\Delta K}$ seems to have 4 possible outcomes (except for the split code for 0-simplex $\{a_i\}$ which has only 2 possible outcomes). However, there exist constraints between these split codes. For example, in Figure 5, the code 1 for 1-simplex id 1 implies that 2-simplex id 1 also has code 1. This in turn implies that 1-simplex id 2 cannot have code 2. Similarly, code 2 for 1-simplex id 3 implies a code 2 for 2-simplex id 2, which in turn implies that 1-simplex id 4 cannot have code 1. These constraints, illustrated in the "scoreboard" of Figure 6, can be summarized using the following two rules:

(1) If a simplex has split code $c \in \{1, 2\}$, all of its parents have split code c.

(2) If a simplex has split code 3, none of its parents have split code 4.

As we encode split codes in $C_i^{\Delta K}$ left to right, we apply these two rules (and their contrapositives) transitively to constrain the possible outcomes for split codes yet to be encoded. Using arithmetic coding with uniform outcome probabilities, these constraints reduce the code string length in Figure 6 from 15 bits to 10.2 bits. In our models, the constraints reduce the code string from 30 bits to 14 bits on average.

The code string is further reduced using a non-uniform probability model. We create an array $T[0..dim][0..15]$ of encoding tables, indexed by simplex dimension (0..dim) and by the set of possible (constrained) split codes (a 4-bit mask). For each simplex s, we encode its split code c using the probability distribution found in $T[s.dim][s.codes\_mask]$. For 2-dimensional models, only 10 of the 48 tables are non-trivial, and each table contains at most 4 probabilities, so the total size of the probability model is small. These encoding tables reduce the code strings to approximately 8 bits as shown in Table 1. By comparison, the PM representation requires approximately 5 bits for the same information, but of course it disallows topological changes.

To provide more intuition for the efficiency of the PSC representation, we note that capturing the connectivity of an average 2-manifold simplicial complex (n vertices, $3n$ edges, and $2n$ triangles) requires $\sum_{i=1}^{n}(\log_2 i+8) \simeq n(\log_2 n+7)$ bits with PSC encoding, versus $n(12\log_2 n + 9.5)$ bits with a traditional one-way incidence graph representation.

For improved compression, it would be best to use a hybrid PM + PSC representation, in which the more concise PM vertex split encoding is used when the local neighborhood is an orientable 2-dimensional manifold (this occurs on average 93% of the time in our examples).

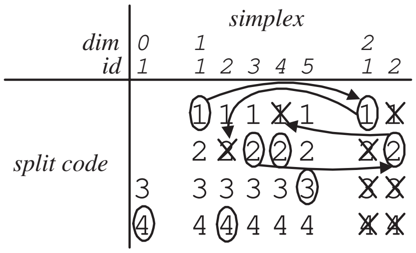


Figure 6: Constraints on the split codes for the simplices in the example of Figure 5.

Table 1: Compression results and construction times.

Object	#verts n	Space required (bits/n)							Trad. repr. bits/n	Con. time hrs.
		K		V		D	A	Σ		
		$\{a_i\}$	$C_i^{\Delta K}$	$midp_i$	$(\Delta\mathbf{v})_i$	$C_i^{\Delta D}$	$C_i^{\Delta A}$			
drumset	34,794	12.2	8.2	0.9	28.1	4.1	0.4	53.9	146.1	4.3
destroyer	83,799	13.3	8.3	0.7	23.1	2.1	0.3	47.8	154.1	14.1
chandelier	36,627	12.4	7.6	0.8	28.6	3.4	0.8	53.6	143.6	3.6
schooner	119,734	13.4	8.6	0.7	27.2	2.5	1.3	53.7	148.7	22.2
sandal	4,628	9.2	8.0	0.7	33.4	1.5	0.0	52.8	123.2	0.4
castle	15,082	11.0	1.2	0.6	30.7	0.0	-	43.5	-	0.5
cessna	6,795	9.6	7.6	0.6	32.2	2.5	0.1	52.6	132.1	0.5
harley	28,847	11.9	7.9	0.9	30.5	1.4	0.4	53.0	135.7	3.5

To compress $C_i^{\Delta D}$, we predict the material for each new principal simplex $s \in \text{star}(\{a_i\}) \cup \text{star}(\{b_i\}) \subset K^{i+1}$ by constructing an ordered set D_s of materials found in $\text{star}(\{a_i\}) \subset K^i$. To improve the coding model, the first materials in D_s are those of principal simplices in $\text{star}(s') \subset K^i$ where s' is the ancestor of s; the remaining materials in $\text{star}(\{a_i\}) \subset K^i$ are appended to D_s. The entry in $C_i^{\Delta D}$ associated with s is the index of its material in D_s, encoded arithmetically. If the material of s is not present in D_s, it is specified explicitly as a global index in D.

We encode $C_i^{\Delta A}$ by specifying the area a_s for each new principal simplex $s \in \mathcal{P}_{01}(\text{star}(\{a_i\}) \cup \text{star}(\{b_i\})) \subset K^{i+1}$. To account for this redistribution of area, we identify the principal simplex from which s receives its area by specifying its index in $\mathcal{P}_{01}(\text{star}(\{a_i\})) \subset K^i$.

The column labeled Σ in Table 1 sums the bits of each field of the *gvspl* records. Multiplying Σ by the number n of vertices in $\hat{M}$ gives the total number of bits for the PSC representation of the model (e.g. 500 KB for the destroyer). By way of comparison, the next column shows the number of bits per vertex required in a traditional "IndexedFaceSet" representation, with quantization of 16 bits per coordinate and arithmetic coding of face materials ($\simeq 3n\cdot 16 + 2n\cdot 3\cdot \log_2 n$ + materials).

4 PSC CONSTRUCTION

In this section, we describe a scheme for iteratively choosing pairs of vertices to unify, in order to construct a PSC representation. Our algorithm, a generalization of [13], is time-intensive, seeking high quality approximations. It should be emphasized that many quality metrics are possible. For instance, the *quadric error* metric recently introduced by Garland and Heckbert [9] provides a different tradeoff of execution speed and visual quality.

As in [13, 20], we first compute a cost ΔE for each candidate *vunify* transformation, and enter the candidates into a priority queue ordered by ascending cost. Then, in each iteration $i = n-1 \ldots 1$, we perform the *vunify* at the front of the queue and update the costs of affected candidates.

4.1 Forming set $\mathcal{C}$ of candidate vertex pairs

In principle, we could enter all possible pairs of vertices from $\hat{M}$ into the priority queue, but this would be prohibitively expensive since simplification would then require at least $O(n^2 \log n)$ time. Instead, we would like to consider only a smaller set $\mathcal{C}$ of candidate vertex pairs. Naturally, $\mathcal{C}$ should include the 1-simplices of K. Additional pairs should also be included in $\mathcal{C}$ to allow distinct connected components of M to merge and to facilitate topological changes. We considered several schemes for forming these additional pairs, including binning, octrees, and k-closest neighbor graphs, but opted for the Delaunay triangulation because of its adaptability on models containing components at different scales.

We compute the Delaunay triangulation of the vertices of $\hat{M}$, represented as a 3-dimensional simplicial complex $\hat{K}_{DT}$. We define the initial set $\mathcal{C}$ to contain both the 1-simplices of $\hat{K}$ and the subset of 1-simplices of $\hat{K}_{DT}$ that connect vertices in different connected components of $\hat{K}$. During the simplification process, we apply each vertex unification performed on M to $\mathcal{C}$ as well in order to keep consistent the set of candidate pairs.

For models in $\mathbf{R}^3$, $\mathcal{C} \cap \mathrm{star}(\{a_i\})$ has constant size in the average case, and the overall simplification algorithm requires $O(n \log n)$ time. (In the worst case, it could require $O(n^2 \log n)$ time.)

4.2 Selecting vertex unifications from $\mathcal{C}$

For each candidate vertex pair $(a, b) \in \mathcal{C}$, the associated *vunify*$(\{a\}, \{b\}) : M^i \leftarrow M^{i+1}$ is assigned the cost

$$\Delta E = \Delta E_{dist} + \Delta E_{disc} + E_{\Delta area} + E_{fold} .$$

As in [13], the first term is $\Delta E_{dist} = E_{dist}(M^i) - E_{dist}(M^{i+1})$, where $E_{dist}(M)$ measures the geometric accuracy of the approximate model M. Conceptually, $E_{dist}(M)$ approximates the continuous integral

$$\int_{\mathbf{p} \in \hat{M}} d^2(\mathbf{p}, M) ,$$

where $d(\mathbf{p}, M)$ is the Euclidean distance of the point $\mathbf{p}$ to the closest point on M. We discretize this integral by defining $E_{dist}(M)$ as the sum of squared distances to M from a dense set of points X sampled from the original model $\hat{M}$. We sample X from the set of principal simplices in K — a strategy that generalizes to arbitrary triangulated models.

In [13], $E_{disc}(M)$ measures the geometric accuracy of discontinuity curves formed by a set of sharp edges in the mesh. For the PSC representation, we generalize the concept of sharp edges to that of *sharp simplices* in K — a simplex is sharp either if it is a boundary simplex or if two of its parents are principal simplices with different material identifiers. The energy E_{disc} is defined as the sum of squared distances from a set X_{disc} of points sampled from sharp simplices to the discontinuity components from which they were sampled. Minimization of E_{disc} therefore preserves the geometry of material boundaries, normal discontinuities (creases), and triangulation boundaries (including boundary curves of a surface and endpoints of a curve).

We have found it useful to introduce a term $E_{\Delta area}$ that penalizes surface stretching (a more sophisticated version of the regularizing E_{spring} term of [13]). Let A_N^{i+1} be the sum of triangle areas in the neighborhood $\mathrm{star}(\{a_i\}) \cup \mathrm{star}(\{b_i\}) \subset K^{i+1}$, and A_N^i the sum of triangle areas in $\mathrm{star}(\{a_i\}) \subset K^i$. The mean squared displacement over the neighborhood N due to the change in area can be approximated as $\overline{disp^2} = \frac{1}{2}(\sqrt{A_N^{i+1}} - \sqrt{A_N^i})^2$. We let $E_{\Delta area} = |X_N| \, \overline{disp^2}$, where $|X_N|$ is the number of points X projecting in the neighborhood.

To prevent model self-intersections, the last term E_{fold} penalizes surface folding. We compute the rotation of each oriented triangle in the neighborhood due to the vertex unification (as in [10, 20]). If any rotation exceeds a threshold angle value, we set E_{fold} to a large constant.

Unlike [13], we do not optimize over the vertex position $\mathbf{v}_a^i$, but simply evaluate ΔE for $\mathbf{v}_a^i \in \{\mathbf{v}_a^{i+1}, \mathbf{v}_b^{i+1}, (\mathbf{v}_a^{i+1} + \mathbf{v}_b^{i+1})/2\}$ and choose the best one. This speeds up the optimization, improves model compression, and allows us to introduce non-quadratic energy terms like $E_{\Delta area}$.

5 RESULTS

Table 1 gives quantitative results for the examples in the figures and in the video. Simplification times for our prototype are measured on an SGI Indigo2 Extreme (150MHz R4400). Although these times may appear prohibitive, PSC construction is an off-line task that only needs to be performed once per model.

Figure 9 highlights some of the benefits of the PSC representation. The pearls in the chandelier model are initially disconnected tetrahedra; these tetrahedra merge and collapse into 1-d curves in lower-complexity approximations. Similarly, the numerous polygonal ropes in the schooner model are simplified into curves which can be rendered as line segments. The straps of the sandal model initially have some thickness; the top and bottom sides of these straps merge in the simplification. Also note the disappearance of the holes on the sandal straps. The castle example demonstrates that the original model need not be a mesh; here $\hat{M}$ is a 1-dimensional non-manifold obtained by extracting edges from an image.

6 RELATED WORK

There are numerous schemes for representing and simplifying triangulations in computer graphics. A common special case is that of subdivided 2-manifolds (meshes). Garland and Heckbert [12] provide a recent survey of mesh simplification techniques. Several methods simplify a given model through a sequence of edge collapse transformations [10, 13, 14, 20]. With the exception of [20], these methods constrain edge collapses to preserve the topological type of the model (e.g. disallow the collapse of a tetrahedron into a triangle).

Our work is closely related to several schemes that generalize the notion of edge collapse to that of vertex unification, whereby separate connected components of the model are allowed to merge and triangles may be collapsed into lower dimensional simplices. Rossignac and Borrel [21] overlay a uniform cubical lattice on the object, and merge together vertices that lie in the same cubes. Schaufler and Stürzlinger [22] develop a similar scheme in which vertices are merged using a hierarchical clustering algorithm. Luebke [18] introduces a scheme for locally adapting the complexity of a scene at runtime using a clustering octree. In these schemes, the approximating models correspond to simplicial complexes that would result from a set of *vunify* transformations (Section 3.3). Our approach differs in that we order the *vunify* in a carefully optimized sequence. More importantly, we define not only a simplification process, but also a new representation for the model using an encoding of *gvspl* = *vunify*$^{-1}$ transformations.

Recent, independent work by Schmalstieg and Schaufler [23] develops a similar strategy of encoding a model using a sequence of vertex split transformations. Their scheme differs in that it tracks only triangles, and therefore requires regular, 2-dimensional triangulations. Hence, it does not allow lower-dimensional simplices in the model approximations, and does not generalize to higher dimensions.

Some simplification schemes make use of an intermediate volumetric representation to allow topological changes to the model. He et al. [11] convert a mesh into a binary inside/outside function discretized on a three-dimensional grid, low-pass filter this function, and convert it back to a simpler surface using an adaptive "marching cubes" algorithm. They demonstrate that aliasing is reduced by rendering the filtered volume as a set of nested translucent surfaces. Similarly, Andújar et al. [1] make use of an inside/outside octree representation.

Triangulations of subdivided manifolds (and non-manifolds) of higher dimension are used extensively in solid modeling. Paoluzzi et al. [19] provide an overview of related work and analyze the benefits of representing such triangulations using (regular) simplicial complexes. Bertolotto et al. [3, 4] present hierarchical simplicial representations for subdivided manifolds, but these do not support changes of topological type.

Polyhedra can also be represented using more general representations. The simplicial set representation of Lang and Lienhardt [17] generalizes simplicial complexes to allow incomplete and degenerate simplices. Cell complexes, formed by subdividing manifolds into non-simplicial cells, can be represented using the radial edge structure of Weiler [29] or the cell tuple structure of Brisson [5].

7 SUMMARY AND FUTURE WORK

We have introduced the progressive simplicial complex representation, a new format for arbitrary triangulated models that captures both geometry and topology in a unified multiresolution framework. It defines a continuous-resolution sequence of approximating models, from the original model down to a single vertex. In addition, it allows geomorphs between any pair of models in this sequence, supports progressive transmission, and offers a concise storage format. We presented an optimization algorithm for constructing PSC representations for computer graphics surface models.

Although we restricted our examples in this paper to models of dimension at most 2, the PSC representation is defined for arbitrary dimensions, and we expect that it will find useful applications in the representation of higher dimensional models such as volumes, light fields, and bidirectional reflection distribution functions. In particular, it offers an avenue for level-of-detail control in volume rendering applications.

ACKNOWLEDGMENTS

We are extremely grateful to Viewpoint Datalabs for providing us with numerous meshes with which to experiment. We also wish to thank Tom Duchamp for helpful discussions on algebraic topology.

REFERENCES

[1] ANDÚJAR, C., AYALA, D., BRUNET, P., JOAN-ARINYO, R., AND SOLÉ, J. Automatic generation of multiresolution boundary representations. *Computer Graphics Forum (Proceedings of Eurographics '96) 15*, 3 (1996), 87–96.

[2] BAJAJ, C., AND SCHIKORE, D. Error-bounded reduction of triangle meshes with multivariate data. *SPIE 2656* (1996), 34–45.

[3] BERTOLOTTO, M., DE FLORIANI, L., BRUZZONE, E., AND PUPPO, E. Multiresolution representation of volume data through hierarchical simplicial complexes. In *Aspects of visual form processing* (1994), C. Arcelli, L. Cordella, and G. Sanniti di Baja, Eds., World Scientific, pp. 73–82.

[4] BERTOLOTTO, M., DE FLORIANI, L., AND MARZANO, P. Pyramidal simplicial complexes. In *Solid Modeling '95* (May 1995), pp. 153–162.

[5] BRISSON, E. *Representation of d-dimensional geometric objects*. PhD thesis, Dept. of Computer Science and Engineering, U. of Washington, 1990.

[6] CLARK, J. Hierarchical geometric models for visible surface algorithms. *Communications of the ACM 19*, 10 (October 1976), 547–554.

[7] DEERING, M. Geometry compression. *Computer Graphics (SIGGRAPH '95 Proceedings)* (1995), 13–20.

[8] FUNKHOUSER, T., AND SÉQUIN, C. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. *Computer Graphics (SIGGRAPH '93 Proceedings)* (1993), 247–254.

[9] GARLAND, M., AND HECKBERT, P. Surface simplification using quadric error metrics. *Computer Graphics (SIGGRAPH '97 Proceedings)* (1997).

[10] GUÉZIEC, A. Surface simplification inside a tolerance volume. Research Report RC-20440, IBM, March 1996.

[11] HE, T., HONG, L., VARSHNEY, A., AND WANG, S. Controlled topology simplification. *IEEE Transactions on Visualization and Computer Graphics 2*, 2 (June 1996), 171–184.

[12] HECKBERT, P., AND GARLAND, M. Survey of polygonal surface simplification algorithms. Tech. Rep. CMU-CS-95-194, Carnegie Mellon University, 1995.

[13] HOPPE, H. Progressive meshes. *Computer Graphics (SIGGRAPH '96 Proceedings)* (1996), 99–108.

[14] HOPPE, H., DEROSE, T., DUCHAMP, T., MCDONALD, J., AND STUETZLE, W. Mesh optimization. *Computer Graphics (SIGGRAPH '93 Proceedings)* (1993), 19–26.

[15] HUDSON, J. *Piecewise Linear Topology*. W.A. Benjamin, Inc, 1969.

[16] JUNGERMAN, M., AND RINGEL, G. Minimal triangulations on orientable surfaces. *Acta Mathematica 145*, 1-2 (1980), 121–154.

[17] LANG, V., AND LIENHARDT, P. Geometric modeling with simplicial sets. In *Pacific Graphics '95* (August 1995), pp. 475–493.

[18] LUEBKE, D. Hierarchical structures for dynamic polygonal simplification. TR 96-006, Department of Computer Science, University of North Carolina at Chapel Hill, 1996.

[19] PAOLUZZI, A., BERNARDINI, F., CATTANI, C., AND FERRUCCI, V. Dimension-independent modeling with simplicial complexes. *ACM Transactions on Graphics 12*, 1 (January 1993), 56–102.

[20] RONFARD, R., AND ROSSIGNAC, J. Full-range approximation of triangulated polyhedra. *Computer Graphics Forum (Proceedings of Eurographics '96) 15*, 3 (1996), 67–76.

[21] ROSSIGNAC, J., AND BORREL, P. Multi-resolution 3D approximations for rendering complex scenes. In *Modeling in Computer Graphics*, B. Falcidieno and T. L. Kunii, Eds. Springer-Verlag, 1993, pp. 455–465.

[22] SCHAUFLER, G., AND STÜRZLINGER, W. Generating multiple levels of detail from polygonal geometry models. In *Virtual Environments '95 (Eurographics Workshop)* (January 1995), M. Göbel, Ed., Springer Verlag, pp. 33–41.

[23] SCHMALSTIEG, D., AND SCHAUFLER, G. Smooth levels of detail. In *Proc. of IEEE 1997 Virtual Reality Annual Intnl. Symp.* (1997), pp. 12–19.

[24] SCHROEDER, W., ZARGE, J., AND LORENSEN, W. Decimation of triangle meshes. *Computer Graphics (SIGGRAPH '92 Proceedings) 26*, 2 (1992), 65–70.

[25] SPANIER, E. H. *Algebraic Topology*. McGraw-Hill, New York, 1966.

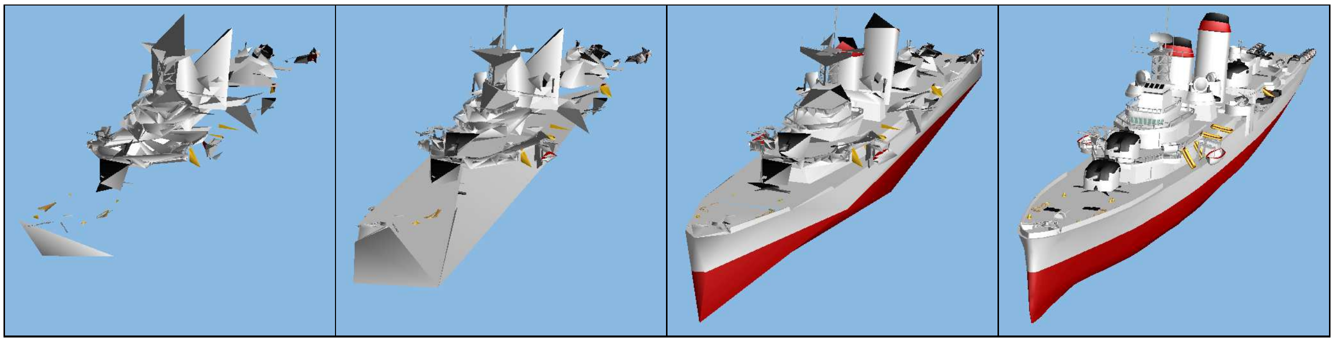
[26] TAUBIN, G., AND ROSSIGNAC, J. Geometry compression through topological surgery. Research Report RC-20340, IBM, January 1996.

[27] TURK, G. Re-tiling polygonal surfaces. *Computer Graphics (SIGGRAPH '92 Proceedings) 26*, 2 (1992), 55–64.

[28] WAVEFRONT TECHNOLOGIES, INC. *Wavefront File Formats, Version 4.0 RG-10-004*, first ed. Santa Barbara, CA, 1993.

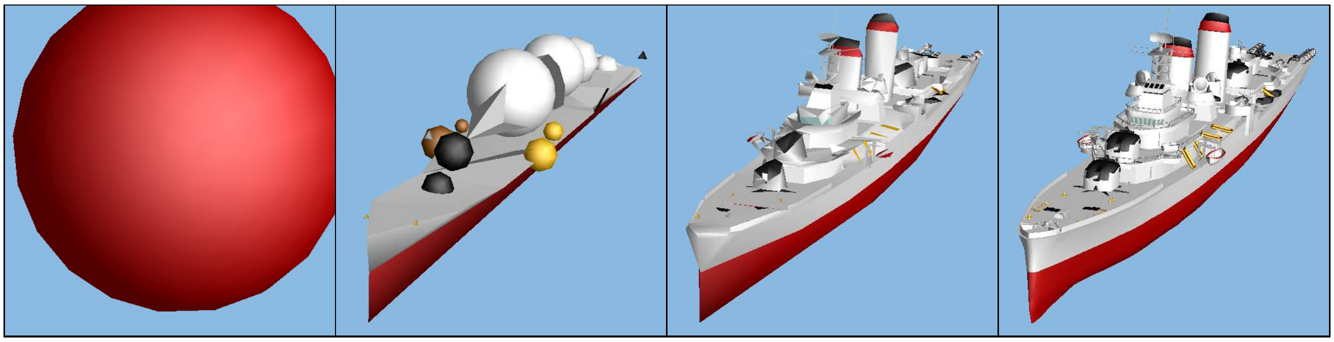
[29] WEILER, K. The radial edge structure: a topological representation for non-manifold geometric boundary modeling. In *Geometric modeling for CAD applications*. Elsevier Science Publish., 1988.

[30] WITTEN, I., NEAL, R., AND CLEARY, J. Arithmetic coding for data compression. *Communications of the ACM 30*, 6 (June 1987), 520–540.



M^0; 1,154 verts; 2,522 tris M^{1739}; 2,893 verts; 6,000 tris M^{2739}; 3,893 verts; 8,000 tris $M^{n=82645}$; 83,799 verts; 167,744 tris

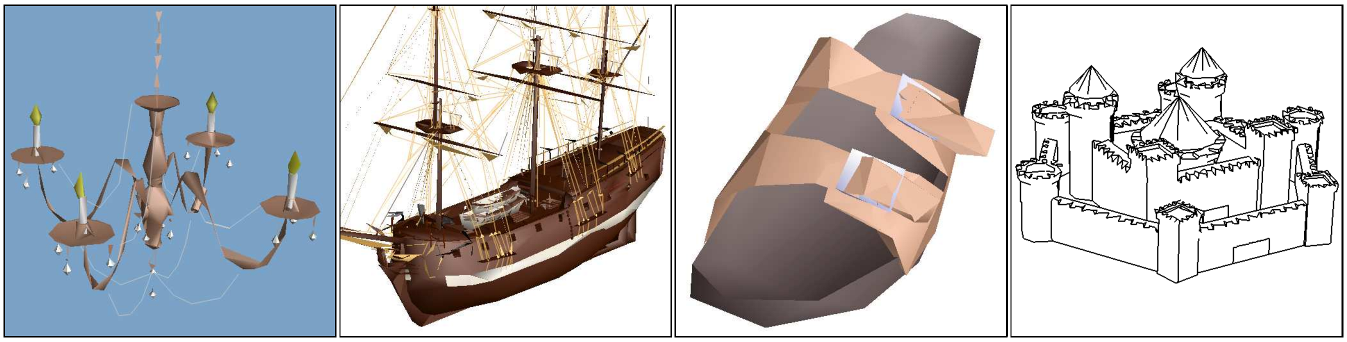
Figure 7: From a given mesh $\hat{M}$, the PM representation [13] captures a sequence of meshes $M^0 \ldots M^n = \hat{M}$. Because all approximations M^i must have the same topological type, the base mesh M^0 may still be complex.



M^1; $\{1, 0, 0\}$; (1) M^{50}; $\{14, 3, 66\}$; (18) M^{1000}; $\{5, 89, 1517\}$; (56) $M^{n=83799}$; $\{0, 0, 167744\}$; (117)

Figure 8: In contrast, the PSC representation captures a sequence of models $M^1 \ldots M^n = \hat{M}$ in which the base model M^1 always consists of a single vertex. All geometric and topological information is encoded progressively by a sequence of generalized vertex split transformations. The image captions indicate the number of principal $\{0, 1, 2\}$-simplices respectively and the number of connected components (in parenthesis). Note that even M^{1000} looks markedly better than the 8000-triangle PM approximation.

$\hat{M}$; 72,346 triangles (276) $\hat{M}$; 232,974 triangles (2154) $\hat{M}$; 8,936 triangles (9) $\hat{M}$; 15,601 segments (39)



M^{500}; $\{3, 52, 674\}$; (50) M^{3000};$\{239, 495, 3189\}$ (587) M^{100}; $\{0, 0, 170\}$; (2) M^{1000}; $\{20, 1265, 0\}$; (33)

Figure 9: For each column, the top row shows the original model and the bottom row shows one approximation in the PSC sequence. The image captions indicate the number of principal $\{0, 1, 2\}$-simplices respectively and the number of connected components (in parenthesis).