A Simple Compression of Tri-Quad Meshes with Handles

Ruben G. Diaz.[1], Marcelo Dreux[1], Hélio Lopes[2], Thomas Lewiner[2]
[1]Deparment of Mechanical Engineering
[2]Department of Mathematics
Pontifícia Universidade Católica do Rio de Janeiro
Rua Marquês de São Vicente, 225, Gávea
Phone: +55 (21) 3527-1162/1639/1164
CEP 22453-900 - Rio de Janeiro - RJ - BRAZIL
rgomez@tecgraf.puc-rio.br, dreux@puc-rio.br, {lopes, lewiner}@mat.puc-rio.br

ABSTRACT

This paper extends a mesh compression method to deal with meshes composed of triangular and quadrilateral faces with genus. The proposed method is very useful given the fact that many CAD meshes have these characteristics. A quad extension has been introduced to the original EdgeBreaker encoding and decoding algorithm. The modification is simple and amounts to a case analysis of the possible compression labels assigned to two triangles originating from splitting a quadrilateral face. The new method entails to augmenting the C,L,E,R,S EdgeBreaker codes with the additional labels c,l,s.

Keywords: Mesh Compression, EdgeBreaker, Geometric Compression, Quad-Tri meshes.

1 INTRODUCTION

In many applications it is necessary to transmit 3D models over the Internet, and these models are usually large and complex. Due to these characteristics it is convenient to compress data in order to transmit them, in a simple and efficient way.

Boundary representation models consist of an indirect representation of a solid or mesh through its border description.

Many boundary representation models for 3D objects have been proposed, most of them represented by triangular meshes. Nevertheless, polygonal meshes with both triangle and quads are used in many CAD applications, asking for a compression scheme of the mesh model.

This work presents a data structure that is a simplification of the *HalfEdge* data structure. This new data structure is called *CHalfedge* and is used for the compression and decompression of static meshes composed by triangles and quads with or without handles.

CHalfedge does not support dynamic meshes, i.e., the mesh cannot be edited or adapted but can be used for progressive meshes transmission [15] [8].

2 RELATED WORK

2.1 Topological Data Structure

There are many well-known data structures used for boundary representation of solids in $\mathbb{R}^3$ [1] [14] [21]. Rantal et al. [16] presented an extension of the *HalfEdge* data structure [14] in order to deal with boundary for cut and paste operations with applications for feature based modeling. In their scheme, the boundary edge is represented as an edge with only one half-edge. *CHalfEdge*, the proposed data structure, uses the same strategy.

Guibas and Stolfi [6] proposed a generalization to include 2-manifolds, orientable or not, where Edge Functions and the corresponding *Edge-Algebra* have been implemented as the *Quad-Edge* data structure with a set of operators.

Dobkin e Lazlo [4] presented the *Facet-Edge* data structure and a set of operators to represent cell complexes that extended the work presented by Guibas and Stolfi. Brisson [2] generalized the last two works to include n-manifolds with the *Cell-Tuple* data structure and operators that are generically called Constructors.

Castelo et al [3] used the Handlebody theory to introduce the *Handle-Edge* data structure in order to represent combinatorial surfaces with boundary. They introduced a set of operators to create handles. Lopes and Tavares [13] presented the *Handle-Face* data structure and a set of operators to represent 3-manifolds.

2.2 Geometric Compression

Many mesh compression methods have been proposed, among them the Topological Surgery compression method proposed by Taubin [20]. This method compresses the TST(Triangle Spanning Tree [20]) tree as well as its dual vertex tree. The MPEG-4 standard for 3D compression is based on the Topological Surgery technique [20], which was co-invented by Rossignac at IBM Institute.

Another compression technique, also proposed by Rossginac, is the EdgeBreaker compression [18] [17] for triangle meshes without handles. It uses the *CornerTable* data structure [12] to represent the connectivity of any manifold triangle mesh by the use of two integer arrays.

Rossignac also showed a decompression technique to triangular meshes that has been codified by the EdgeBreaker method. This decompression method is known as Wrap&Zip [19] and it has a quadratic compression in the worst case. The method presented by Rossignac is restricted to closed meshes, homeomorphic to a sphere, without boundary and without genus.

King et al [9] presented a codification and compression for quad meshes homeomorphic to a sphere. Although in this work the authors have mentioned that it is possible to extend the algorithm to surfaces with handles using irregular meshes (triangles and quads), they do not present an algorithm.

Lopes et al [12] extended the work of Rossignac [18] [17] of triangle meshes considering handles in a *CornerTable* data structure and Lewiner presented a codification for meshes with an arbitrary topology [11]. Lage et al [10] also uses a pseudo-representation of the *HalfEdge* data structure, like the representation used in this work, in order to create a scalable topological data structure only for triangle and tetrahedral meshes.

3 BASIC CONCEPTS

In this section some basic concepts of geometry and topology are presented [5]. In the m-dimensional Euclidean vector space, a *convex cell* $\sigma \in \mathbb{R}^m$ is a compact set (limited and closed), not empty, that is the solution of a finite set of linear equalities $f_i(v) = 0$ and linear inequalities $g_i(v)$ where $v = (x_1, x_2, ..., x_m) \mapsto \lambda_0 + \lambda_1 x_1 + \lambda_2 x_2 + ... + \lambda_m x_m = 0$.

A face, or a subcell of a cell σ, is obtained replacing some inequalities $g_i \geq 0$ by equalities. A vertex of a cell is a subcell with only one point. The cell σ is considered a subcell of itself. A subcell of σ that is not itself is called proper subcell. A *simplex* of dimension n is the convex hull of $n+1$ linearly independents points in $\mathbb{R}^m$.

The subcells of cell σ are defined by σ itself and not by the choice of a particular equations and inequations that represent them. This can be inferred by the following propositon:

All τ cells are a subcell of σ if and only if:

1. If P is the hyperplane generated by τ, then $P \cap \sigma = \tau$
2. No point of P belongs to a convex combination of two points of $\sigma - \tau$.

A n-dimensional complex cell K is a finite collection of i-dimensional cells $(i = 0, ..., n)$ in $\mathbb{R}^m$ under the following conditions:

1. If $\sigma \in K$ and $\tau < \sigma$ then $\tau \in K$.
2. If σ and $\rho \in K$ then σ e ρ are properly linked.

The 0, 1 and 2-cells of the complex K are called, respectively, the *vertices*, *edges* and *faces* of K.

The *underlying polyhedron* $|K| \in \mathbb{R}^m$ corresponds to the union of the cells in K. If a collection of cells $L \in K$ is a cell complex, then it is called a subcomplex of K.

A complex K is *connected* if it cannot be represented as a union of two non empty disjoint subcomplexes L and M without common cells. A *component* of a simplicial complex K is a connected subcomplex that is not contained in a larger connected subcomplex of K.

A combinatorial n-manifold is *orientable* when it is possible to choose a coherent orientation of its n-cells, i.e., two adjacent n-cells induce opposite orientations on their common $(n-1)$-cells. From now on, a *surface* and a *curve* will always mean respectively, a 2 and 1 dimensional connected oriented combinatorial manifold with or without boundary. The set of faces (2-cells), edges (1-cell) and vertices of a surface will be denoted $F(S)$, $E(S)$ and $V(S)$.

The mesh compression and decompression processes that are presented in this article are applied to combinatorial oriented surfaces without boundary. So it is necessary to respect some characteristics that ensures the preservation of the topological combinatorial aspects of the mesh. The next definition grants the existence of such characteristic for combinatorial surfaces:

Let NV, NE, e NF be respectively, the number of vertices, edges and faces of a combinatorial surface $\mathscr{S}$.

$$\chi(\mathscr{S}) = NV - NE + NF \qquad (1)$$

Then the sum is a constant for all surfaces with the same topological type of $\mathscr{S}$. This constant is called Euler characteristic.

This definition can be expanded to include other components of the model.Consider an orientable surface S as a single connected component and let g be the number of genus and b the number of boundary curves of the surface. Then, the Euler-Poincaré formula is written as:

$$\chi(\mathscr{S}) = NV - NE + NF = 2 - 2g - b \qquad (2)$$

The number of Euler-Poincaré $\chi(\mathscr{S})$ defines univocally the surface S. When the surface is oriented and has no boundary, the number of genus g of S, knowing only the number of vertices, edges and faces of $S(NV,NE,NF)$ can be obtained by the Euler-Poincaré formula:

$$NF - NE + NV = 2 - 2g \tag{3}$$

$$\Downarrow$$

$$g = 1 - \frac{NF}{2} + \frac{NE}{2} - \frac{NV}{2} \tag{4}$$

The attachment of handles to a n-manifold component is relevant to a complete representation of boundaries, and is based on the HandleBody Theory [3] and [12].

4 CHALFEDGE DATA STRUCTURE

The CHalfEdge data structure proposed in this work is a simplification of the HalfEdge [14] data structure. It is suitable for polygonal meshes composed by triangles and quads and also uses the concept of half-edge that represents the association of a face with one of its boundary edges. The advantage of this simplification is that it works with two arrays of integers, V and M, which are later described.

Consider an oriented combinatorial surface S without border having NT triangles, NQ quads, NE edges and NV vertices. In this data structure the half-edges, the vertices and the faces are indexed by non-negative integers.

Each triangle is represented by 3 consecutive half-edges that define its orientation. Analogously, a quad is represented by four consecutive half-edges. For example, half-edges 0, 1 and 2 correspond to the first triangle; half-edges 3, 4 and 5 corresponds to the second triangle, half-edges 6, 7, 8 and 9 correspond to the first quad if there is a quad in the mesh and so on. A half-edge with index $h \in [0, 3NT - 1]$ belongs to a triangle, and if $h \in [3NT, 3NT + 4NQ - 1]$ belongs to a quad.

The border of each triangle is implicitly represented by an oriented cycle of 3 half-edges. In the same way, an oriented cycle of 4 half-edges defines the border of a quad. In such a way, the surface S will have $3NT + 4NQ$ half-edges. Assuming a counter-clockwise orientation, the next convention is adopted to index the half-edges of S:

- The indices of the half-edges that form the border of a triangle of index t are: $3t$, $3t + 1$, and $3t + 2$.
- The indices of the half-edges that form the border of a quad q are: $3NT + 4q$, $3NT + 4q + 1$, $3NT + 4q + 2$, and $3NT + 4q + 3$.

The index of a triangle that has a half-edge h is $h.f = h \div 3$. If the half-edge h belongs to a quad, its index is $h.f = (h - 3NT) \div 4$. The next(h.n) and previous (h.p) half-edges of h.f are obtained by the use of the following expressions:

Triangles

$$h.n = 3 * h.f + (h + 1) \quad \text{mod } 3 \tag{5}$$

$$h.p = 3 * h.f + (h + 2) \quad \text{mod } 3 \tag{6}$$

Quads

$$h.n = 3 * NT + 4 * h.f + (h + 1) \quad \text{mod } 4 \tag{7}$$

$$h.p = 3 * NT + 4 * h.f + (h + 3) \quad \text{mod } 4 \tag{8}$$

The CHalfEdge data structure represents the geometry of a surface S by the association of each half-edge h with its geometrical vertex index h.v. The edge adjacency between the neighboring triangles or quads of S is represented by the association of each half-edge h with its opposite half-edge h.m or M[h](mate), which has the same opposite geometrical edge(formally M[M[h]] = h, see Figure 1). The geometry and adjacency information are stored in two integer arrays, called the V and M tables, respectively.

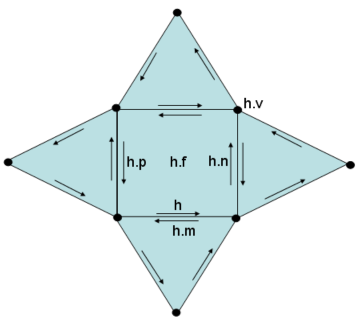
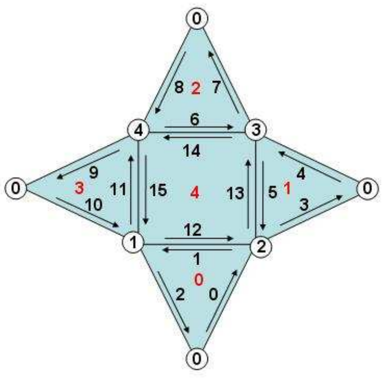
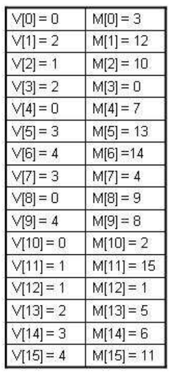


Figure 1: CHalfEdge notations.

According to the definition of combinatorial surface without bounding border, each edge is shared by only 2 faces. Two adjacent faces that share the same edge have half-edges with opposite orientation.

To illustrate the data structure tables consider the pyramid of Figure 2. When the half-edge h is associated to a boundary edge, its M[h] value is assigned to -1.





V[0] = 0	M[0] = 3
V[1] = 2	M[1] = 12
V[2] = 1	M[2] = 10
V[3] = 2	M[3] = 0
V[4] = 0	M[4] = 7
V[5] = 3	M[5] = 13
V[6] = 4	M[6] =14
V[7] = 3	M[7] = 4
V[8] = 0	M[8] = 9
V[9] = 4	M[9] = 8
V[10] = 0	M[10] = 2
V[11] = 1	M[11] = 15
V[12] = 1	M[12] = 1
V[13] = 2	M[13] = 5
V[14] = 3	M[14] = 6
V[15] = 4	M[15] = 11

Figure 2: A pyramid surface with its CHalfEdge tables.

Table V of Figure 2 indicates that half-edge 0 has vertex 0, half-edge 1 has vertex 2, and so on. Table M indicates that half-edge 0 has as its correspondent mate, or opposite, half-edge 3, half-edge 1 has its correspondent mate half-edge 12, and so on.

5 COMPRESSION OF TRI-QUAD MESHES

The algorithm for compression and decompression of tri-quad meshes proposed in this work is a generalization of the EdgeBreaker algorithm introduced by Rossignac [17].

The EdgeBreaker is a state machine that encodes a combinatorial surface without boundary S. At each state, a decision is made to move from a triangle Y to an adjacent triangle X. To perform this decision, all visited triangles and their incident vertices are marked.

Let Left and Right denote the other two triangles that are incident upon X. Let v be the vertex common to X, Left and Right. There are five possible situations denoted by the letters C,L,E,R and S (arranged for the mnemonic "Claire's"). The arrows in Figure 3 indicate the direction to the next triangle. Previously visited triangles are filled. The edge that crosses from Y to X is called the gate of one triangle to another.

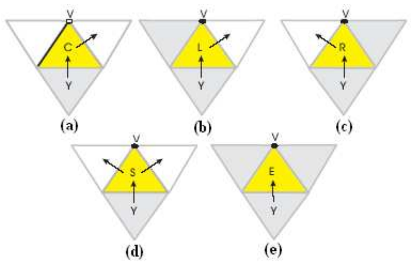


Figure 3: EdgeBreaker clers cases.

The EdgeBreaker algorithm walks to the right whenever is possible(C or L labels, see Figure 3a and 3b respectively). When the right triangle has already been visited it walks to the left(R label - Figure 3c). A triangle gets label S (split - Figure 3d) if the vertice v is already visited and both right and left triangles are not visited yet. In this case a branch in the dual spanning tree is created. Each of these branches has to be traversed and ends up with a label of type E (End - 3e). Whenever the traverse of a S triangle reaches the end of the right branch, the algorithm goes back to that S triangle and then traverses the left branch.

This kind of surface traversing is always possible when the surface is homeomorphic to a sphere. When the surface without boundary has g genus, there are some cases that when going back to the beginning of the branch generated by a label of type S, the left face of this branch has already been visited during the right traversing. Lopes et al [12], proposed in these cases a correct reconstruction by transmitting the edge to the left of the S triangle.

Lopes et al [12], proved that for a surface without border and with g genus, there are $2g$ S triangles of this type. Then the number $2g$ exactly corresponds to the number of handle operations of type 1 (see [12]) that must be applied to the surface during the decompression process to obtain its topology correctly.

5.1 EdgeBreaker Quad Extension

A simple way to compress a quad mesh is to divide each polygon into two triangles and apply the EdgeBreaker algorithm for triangle meshes. But, in the decompression process it is necessary to recognize which faces have been added in order to reconstruct the original mesh(see Figure 4).

King et al [9] suggest to divide a quad into two adjacent triangles, by implicitly adding a diagonal edge. This division is done in such way to force the second triangle to be always to the right of the first triangle. The quad subdivision rules also guarantee that the first triangle label is never R or E because the next triangle of the quad has not been visited. Furthermore, if the first triangle label is of type C, the second label cannot be of type L or E(see Figure 4).

For a triangle there are 5 possible combinations (see EdgeBreaker compression). For other types of polygons the number of combinations can be calculated by the recurrence relationship, based on the fact that a vertex that has not yet been visited can not be incident to an edge previously visited. The solution of this recurrence for polygon with n edges is the Fibonacci number $F(2n-1)$. Then for a quad there are $F(7) = 13$ possible combinations of edges and vertices not adjacent to a visited and not visited gate.

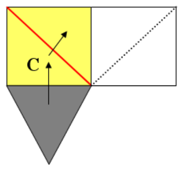


Figure 4: Quad division proposed by King [9].

The approach division leads to 13 possible pairs of label combinations for a quad: CC, CR, CS, LC, LL, LE, LR, LS, SC, SL, SE, SR and SS. In order to build the dual spanning tree, the original EdgeBreaker rules are used for each triangle generated by a quad subdivision.

5.2 Tri-Quad Mesh Codification

The algorithm developed in this work encodes an arbitrary surface by generating 3 separated information which are used to transmit a mesh over the Internet: a sequence of CLERS containing the labels c, l, s, C, L, E, R, and S of each codified face of the surface; the geometry of the vertices and the handles. Thus, in order to compress a mesh composed by triangles and/or quads, must follow these rules:

1. If the face is a triangle, the labels C, L, R, E, or S are used, as in the original EdgeBreaker;
2. If the face is a quad, it is split into two triangles according to King's rule, and the first triangle is encoded with labels c, l or s. The second triangle receives labels C, L, R, E, or S.

The classification of the first quad triangle as c, l or s is the same as the original EdgeBreaker classification. The second quad triangle is always the exit of the quad. The use of a lower case label indicates that the next polygon is a quad.

Lopes et al [12] already mentioned approach is used when dealing with special cases of S triangles. In the next section it will be proved that even for surfaces with triangles and quads the number of transmitted edges is also 2g.

5.3 Analysis of the Compression Method

The compression algorithm creates the spanning tree of the dual graph, where graph nodes are the faces and graph lines represent adjacency between faces. It has complexity O(n), where n is the number of faces.

The algorithm traverses all the faces classifying them. Each face is visited only once, so the spanning tree in the dual graph has $NT + NQ - 1$ lines that correspond to the edges that have gates from one face to another.

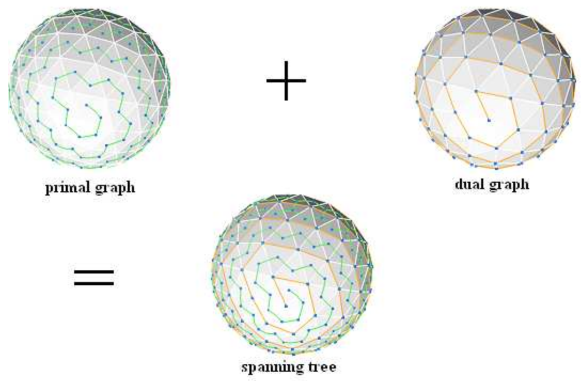


Figure 5: Compression in the primal and dual graph of the spanning tree.

Each visited face classified as c or C visits a new vertex. All vertices are visited so the total number of c or C labels is equal to NV minus the number of vertices of the initial face that can be 3 or 4 depending on being a triangle or a quad.

The algorithm creates a spanning tree in the primal graph of the surface, where the nodes of the graph are surface vertices and the lines of the graph are surface edges. The lines of the spanning tree of the dual graph correspond to the left edges of triangles labeled c or C and all the edges of the initial faces, except the gate. The number of spanning tree edges of the primal graph is, by definition $NV - 1$ (see Figure 5).

The edges implicitly encoded by the algorithm are: the edge gates of the dual graph and the edges of the primal graph spanning tree. The total number of these edges is:

$$(NT + NQ - 1) + (NV - 1) = NV + NT + NQ - 2 \quad (9)$$

The Euler-Poincare formula is:

$$NV - NE + (NT + NQ) = 2 - 2g \quad (10)$$

So, the surface's total number of edges is defined as:

$$NE = NV + NT + NQ - 2 + 2g \quad (11)$$

By comparing equations (9) and (10), the number of edges that has not been implicitly codified results exactly $2g$, which matches Lopes et al demonstration for triangular meshes [12].

5.4 Codification Cost

According to Euler-Poincaré formulation, a mesh with genus complies with the equation (10), where the number of edges, NE, is $(3NT + 4NQ) \div 2$ and the number of faces, NF, is $(NT + NQ)$

After some simple algebraic manipulations you can reach the following equation:

$$NT + 2NQ = 2NV - 4 + 4g \quad (12)$$

As already mentioned, eight symbols are used (c, l, s, C, L, E, R, and S) in order to codify and decodify tri-quad meshes. Hence, the number of necessary bits to represent a CLERS sequence is 3, since $2^3 = 8$ different numbers could be generated.

Triangles and quadrangles are coded with one and two CLERS labels, thus needing 3 and 6 bits, respectively.

To a tri-quad mesh the total number of bits is then (NT + 2NQ) * 3 bits. By substituting this expression in equation 12, the cost, in bits, to codify a tri-quad mesh without genus is:

$$cost = (2NV - 4) * 3bits \quad (13)$$

Therefore, the cost is less than or equal to 6 bits per vertex.

By the same token, the cost of a tri-quad mesh with genus is:

$$\begin{aligned} cost = (NT + 2NQ) * 3bits + \\ 2g * [log_2(3NT + 4NQ)] * bits \end{aligned} \quad (14)$$

As $NT + 2NQ = 2NV - 4 + 4g$, it follows that:

$$cost = (2NV - 4 + 4g) * 3bits + \\ 2g * [log_2(3NT + 4NQ)] * bits \quad (15)$$

When g/v « 1, which is usually the case, the cost per vertex tends to be equal 6.

6 DECOMPRESSION OF TRI-QUAD MESHES

Last section presented the idea of an algorithm to compress tri-quad meshes with handles. This section will introduce the mechanism to decompress meshes that have been codified with the method presented earlier. The decompression method builds the V and M array of CHalfEdge table by reading the stored CLERS, Geometry and Handles data.

6.1 EdgeBreaker Decompression

The methodology Wrap&Zip to decompress meshes presented in [19] and the strategy adopted by the algorithm Spirale Reversi presented in [7] will be used in this work to perform tri-quad mesh decompression. In order to rebuild a mesh codified by the algorithm presented in the last section, two steps are necessary.

The first step called Wrap decode and use the labels saved in CLERS codification to decide where a new triangle or quad will be added to the mesh that is being reconstructed. The result is a connected topological mesh that corresponds to the TST(triangle spanning tree) of the original mesh.

The second step of decompression is the Zip process, which binds the boundary curves that are not connected in order to end the reconstruction of connectivity and geometry of the surface. To correctly link the boundary edges of the mesh, the Zip process orients the free edges in counter clockwise orientation with labels of type L, R and E. Clockwise is used for labels of type C (See Figure 6).

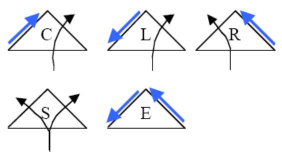


Figure 6: Edges orientation in the Wrap&Zip process.

The decompression process presented in this work also use two stacks as data structure in order to be able to perform a depth search and correctly rebuild the mesh without changing its topology. There is one stack for S labels and another for C labels.

The S stack indicates the left TST branch to be decompressed. The C stack is responsible for the Zip process, by considering the labels that have free edges after the Wrap process. The first element of the C stack will be the last to be decoded.

The Wrap&Zip process decodes and rebuilds the mesh in the same order that the triangles have been visited and coded by the compression process. This kind of decodification has the complexity time in the worst case $O(n^2)$, because this process needs to firstly search the encapsulated sub-sequences of labels S and E, to be able to correctly decode them.

The Spirale Reversi process decodes a mesh by reading the CLERS sequence in reverse order. In this case, it is not necessary to search encapsulated subsequences of labels S and E. This kind of decodification is done in linear time.

The decodification process is similar to the one presented in [7], but CLERS are read in the same order they are generated. Each triangle or quad of type C or c, is saved into a stack, and thus guaranteeing a backward reconstruction, i.e. a pseudo Spirale Reversi method in the Zip decompression process.

The decodification process is not a recursive algorithm because it depends on the data saved in S and C stacks.

7 RESULTS

It is shown some meshes (triangles/quad faces) used to perform the compression and decompression process with the corresponding CLERS.

For each encoded/decoded face it is applied a texture to show the type of CLERS label assigned to it. If a triangle is encoded the label textures show in Figure 7 are used:

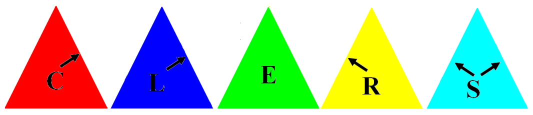


Figure 7: Triangles CLERS textures.

If the encoded/decoded element is a quad, it is divided into two adjacent triangles, as mentioned in section 6, so that there are three cases to consider for the first adjacent triangle when performing a split quad. These triangles are represented by the following label textures shown in Figure 8.

The second adjacent triangle of the quad is codified as if it were a normal triangle. The presented algorithm is able to encode/decode irregular meshes (triangles/quads) with or without an arbitrary number of genus.

Table 1 show some meshes used by compression algorithm proposed in this work, with their respective number of vertices, faces and handles.

Table shows CLERS frequency labels for triangles and quad meshes of table 1.

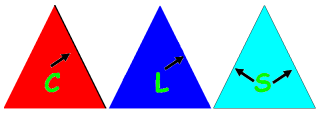


Figure 8: Quad CLERS textures.

Model	NV	NT	NQ	Handles
Bunny	4838	9672	0	0
Fandisk	6475	12946	0	0
cad1	12415	8688	8027	9
cad2	5215	6418	2014	5
knot	1997	0	1996	0

Table 1: Mesh models used to perform compression using CHalfEdge data structure

Model	Bunny	Fandisk	Cad1	Cad2	Knot
#C	4835	6472	4263	3210	0
#L	78	75	91	124	0
#E	377	170	151	254	0
#R	4005	6059	3924	2579	0
#S	376	169	154	250	0
#cC	0	0	422	260	400
#cR	0	0	7164	1372	1191
#cS	0	0	11	13	0
#lC	0	0	66	94	0
#lL	0	0	14	16	0
#lE	0	0	18	18	1
#lR	0	0	88	60	6
#lS	0	0	22	1	0
#sC	0	0	1	3	0
#sL	0	0	6	7	2
#sE	0	0	187	163	394
#sR	0	0	6	7	0
#sS	0	0	0	0	0

Table 2: CLERS frequency labels for tri-quad meshes

Figures 9, 10, 11, 12 and 13 show the meshes with triangles/quad encoded with the proposed method, presented in tables 1 and 2.

8 CONCLUSIONS

In conclusion, we have extended the simple implementation of EdgeBreaker to the connectivity graphs of triangle and quad meshes that represent topological manifolds with handles.

The compression and decompression algorithms use a simple data structure to represent the incidence and adjacency relationship of meshes with triangles and quads.

In addition, the use of an extra data structure (stack structure) in the compression and decompression algorithms enables to encode and decode the CLERS string in linear time, and there is also no need to perform search for encapsulated CLERS subsequences generated by the labels S and E. The decompression is done by a pseudo Spirale Reversi algorithm due to the use of the stack data structure.

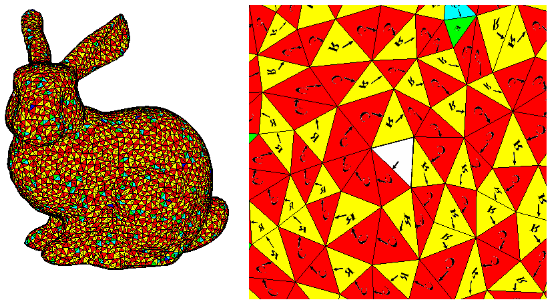


Figure 9: Mesh Bunny with 9672 triangles faces. On the right a detail of the generated compression traversal over the mesh.

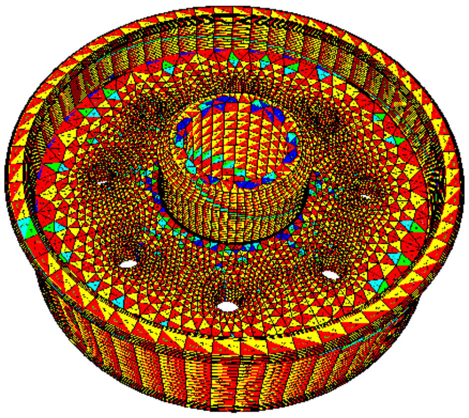


Figure 10: Cad1 mesh with 8688 triangles and 8027 quad faces with 9 handles.

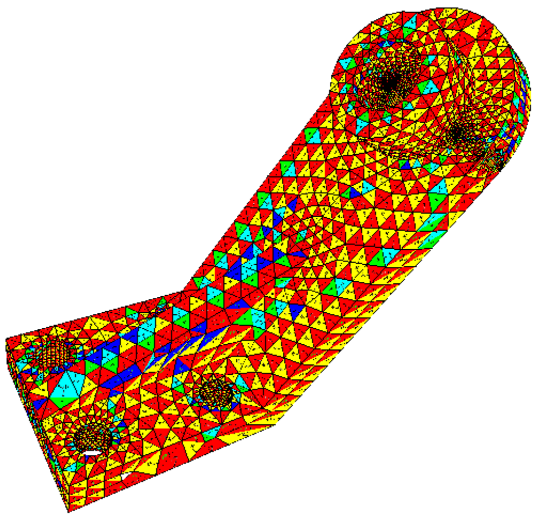


Figure 11: Cad2 mesh with 6418 triangles and 2014 quad faces, with 5 handles.

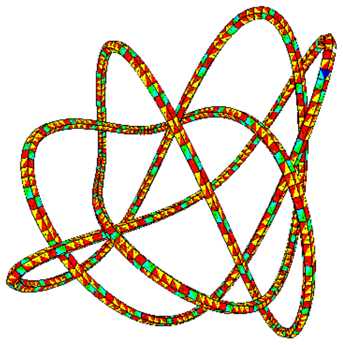


Figure 12: Knot mesh with 1996 quads.

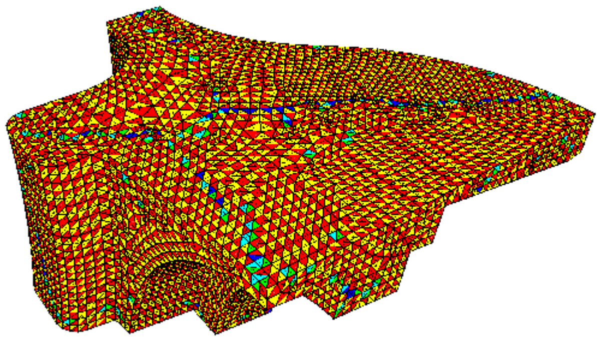


Figure 13: Fandisk mesh with 12946 triangles.

9 ACKNOWLEDGEMENT

The authors are grateful to ICAD/IGames/Vision Lab - the Intelligent CAD and Games Laboratory of PUC-Rio and MatMidia Laboratory of PUC-Rio. The first author is also grateful to CNPq, the Brazilian government research council which partially sponsored this research.

REFERENCES

[1] B. G. Baumgart. A polyhedron representation for computer vision. *AFIPS Proceedings*, (44):119–35, 1975.

[2] E. Brisson. Representing geometric strucutures in d dimesions:topology and order. *Discrete and Computational Geometry*, (9):387–426, 1993.

[3] A. Castelo, H. Lopes, and G. Tavares. Handlebody representation for surfaces and morse operations. *Proceedings on Curves and Surfaces in Computer Vision III*, pages 270–283, 1992.

[4] D.P. Dobkin and M. J. Lazlo. Primitives for the manipulation of three dimensional subdivisions. *Algorithmica*, (4):3–32, 1989.

[5] H. Edelsbrunner, M. J. Ablowitz, S. H. Davis, E. J. Hinch, A. Iserles, J. Ockendon, and P. J. Olver. *Geometry and Topology for Mesh Generation (Cambridge Monographs on Applied and Computational Mathematics)*. Cambridge University Press, New York, NY, USA, 2006.

[6] L.J. Guibas and J. Stolfi. Primitives for the manipulation of general subdivision and the computation of voronoi diagrams. *ACM Transactions on Graphics*, (4):74–123, 1985.

[7] M. Isenburg and S. Snoeyink. Spirale reversi: reverse decoding of the edgebreaker encoding. *Computational Geometry: Theory and Applications*, pages 39–52, 2001.

[8] F. Kälberer, K. Polthier, and C. Tycowicz. Lossless compression of adaptive multiresolution meshes. In *22th Brazilian Symposium on Computer Graphics and Image Processing*, 2009.

[9] D. King, J. Rossignac, and A. Szmczak. Connectivity compression for irregular quadrilateral meshes. *GVU Tech Report GVU-GIT-99-36*, pages 1–21, 2000.

[10] M. Lage, H. Lopes, T. Lewiner, and L. Velho. Chf: A scalable topological data structure for tetrahedral meshes. *18th Brazilian Symposium on Computer Graphics and Image Processing*, pages 349–356, 2005.

[11] T. Lewiner, H. Lopes, J. Rossignac, and A. W. Vieira. Efficient edgebreaker for surfaces of arbitrary topology. *17th Brazilian Symposium on Computer Graphics and Image Processing*, pages 218–225, 2004.

[12] H. Lopes, J. Rossignac, A. Safanova, A. Szymczak, and G. Tavares. Edgebreaker: A simple implementation for surfaces with handles. *Computers&Graphics*, 27(4):553–567, 2003.

[13] H. Lopes and G. Tavares. Structural operators for modelling 3-manifolds. *Fourth ACM Siggraph Symposium on Solid Modelling*, pages 10–18, 1997.

[14] M. Mantyla. *An Introduction to Solid Modelling*. Computer Science Press, RockVille,MD, 1998.

[15] Massimiliano B. Porcu, Nicola Sanna, and Riccardo Scateni. Efficiently keeping an optimal stripification over a clod mesh. *WSCG (Journal Papers)*, pages 73–80, 2005.

[16] M. Rantal, M. Inui, F. Kimura, and M. Mantyla. Cut and paste based modelling with boundary features. *Proceedings of the second ACM/Siggraph Symphosium on Solid Modelling*, pages 303–312, 1993.

[17] J. Rossignac. Edgebreaker: Connectivity compression for triangle meshes. *IEEE Transactions on Visualization and Computers Graphics*, 5(1):47–61, 1999.

[18] J. Rossignac, A. Safanova, and A. Szymczak. A 3d compression made simple: Edgebreaker on a corner-table. *Proceedings of the Shape Modeling International Conference*, pages 278–283, 2001.

[19] J. Rossignac and A. Zzymczak. Wrap&zip decompression of the connectivity of triangle meshes compressed with edgebreaker. *Computational Geometry: Theory and Applications*, 14(1-3):119–135, 1999.

[20] G. Taubin and J. Rossignac. Geometric compression through topological surgery. *ACM Transactions on Graphics*, 17(2):84–115, 1998.

[21] K. J. Weiler. *Topological Structures for Geometric Modeling*. PhD thesis, Rensselaer Polytechnic Institute, 1986.