

HIERARCHICAL CODING OF 3D MODELS WITH SUBDIVISION SURFACES

Francisco Morán and Narciso García

Grupo de Tratamiento de Imágenes, E.T.S. Ing. Telecomunicación
Universidad Politécnica de Madrid — E-28040 Madrid, Spain

E-mail: {fmb,ngs}@gti.ssr.upm.es

ABSTRACT

We build on state of the art methods for multiresolution embedded coding of images, such as Said and Pearlman's Set Partitioning In Hierarchical Trees, and combine them with ideas for 3D objects modelling with subdivision surfaces, to obtain a new technique for hierarchical 3D model coding. The compression ratios we obtain are better than or similar to previously reported ones but, perhaps more importantly, the truly hierarchical coding of 3D objects we propose allows their efficient multiresolution animation. This kind of technique could have a major impact on VRML and MPEG-4, the two ISO standards that now deal with the coding of 3D objects, which are in both cases static and linearly approximated by polygonal meshes. In future versions of those standards, that will have to address the coding of dynamic 3D objects, these will most likely be modelled with higher order primitives such as subdivision surfaces.

1. INTRODUCTION

The simplest way to approximate a 3D surface is to tile it with flat polygons (usually triangles, since triangles are the simplest polygons). Most CAD/CAM applications support polygonal meshes, and the two ISO standards for 3DMC (3D Model Coding), VRML and MPEG-4, are based on them. But polygonal meshes are only piecewise linear approximations to arbitrarily complex surfaces and can thus lead to unacceptable approximation errors unless their number of elements is arbitrarily large. One may well end up having to deal with meshes of hundreds of thousands of facets, which can be too expensive to handle or store — let alone transmit... Furthermore, the animation of polygonal meshes is cumbersome, since their vertices are completely unrelated, and must be moved individually.

The most widely used primitives for modelling 3D objects, especially animated ones, are probably still curved patches, usually from the NURBS (Non-Uniform Rational B-Spline) family. Patches make it easy to generate piecewise polynomial/rational smooth surfaces from relatively

small sets of control points, whose movement deforms locally the surface. But one cannot model objects of arbitrary topology with them without having to face tough patch stitching and curve trimming problems.

Subdivision surfaces [8] are defined as the limit of a recursive refinement process of both the connectivity and the geometry of a control mesh by splitting each of its polygons into several ones. The successive control meshes obtained during this process are inherently nested, and therefore very useful to create multiresolution models of 3D objects, because they define a pyramid of LODs (Levels Of Detail) highly suitable for progressive coding. Unlike the ones of NURBSs, the control meshes of subdivision surfaces can have any topology at no extra cost and, nevertheless, if the subdivision method is carefully designed, the control mesh converges to a limit surface that is as smooth as a piecewise polynomial/rational one (in fact, subdivision surfaces can generalise NURBSs).

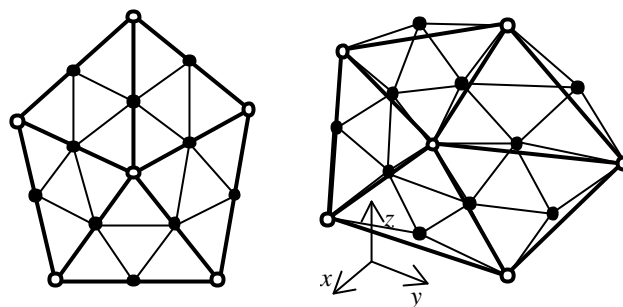


Figure 1 : First two generations of a triangular control mesh subdivided according to an interpolatory scheme.

Left: abstract graph; right: spatial mapping.

Several subdivision schemes have been proposed and studied. For interpolatory schemes (cf. Figure 1), the limit surface interpolates all vertices of all control meshes, while for non-interpolatory ones, control points need not lie on the limit surface. Both kinds of schemes could be useful for the purposes of LODs generation and hierarchical mesh coding, although we focus our study on the interpolatory butterfly scheme devised by Dyn et al. [1].

2. TRULY HIERARCHICAL 3D MODEL CODING

A set of LODs is frequently extracted from a given, very fine mesh, by performing a series of edge collapses or vertex removals, that yield coarser and coarser approximations to the initial, finest LOD. During the reconstruction phase, the finest mesh is recovered from the coarsest one by performing a series of operations that are the inverse of the coarsening phase ones: vertex splits for edge collapses, and vertex insertions for vertex removals. Both these approaches certainly produce *progressive* meshes, in the sense that the finest mesh can be progressively recovered from the coarsest one, but their LODs are not hierarchically nested: they do not form a pyramid of control meshes, as the ones yielded by a subdivision approach do.

Other than the prior art on what we call “*truly hierarchical*” 3DMC and review below, we must mention as well the work on progressive 3DMC already reported: the MPEG-4v2 (MPEG-4 version 2) standard [4] has tools for the efficient progressive coding of static polygonal meshes based on edge collapses and vertex splits; the technique based on vertex removals and insertions by Li and Kuo [5] has the advantage of yielding a fully embedded code.

One of the properties of subdivision methods is that the surfaces associated to the successive control meshes are all homeomorphic (i.e., topologically equivalent) to the surface represented by the base mesh. Unfortunately, as two homeomorphic surfaces can be represented by meshes of very different vertex connectivities, it is rarely straightforward to extract a base control mesh from a given target mesh by making four-to-one merges of its triangles. In fact, the target mesh has usually to be remeshed to have it follow the simple subdivision connectivity rules, but automatic procedures to do it have been described [2].

The problem of automatic LODs extraction in a subdivision surfaces based approach is then, basically, that of adequately remeshing the given target mesh. Once a base control mesh has been extracted from the (remeshed) given target mesh, the subdivision scheme can be repeatedly applied upon it some number of times in the hope of recovering the target mesh. Of course, the new vertices appearing at each stage of the subdivision process do not usually lie on the target surface, but their positions can be corrected after the standard subdivision rules have misplaced them, and before the resulting mesh is used as the input for the next step of the process.

It is easy to imagine then a hierarchical 3D mesh transmission scenario, in which an initial, coarse mesh has already been transmitted somehow and is taken as the base mesh for the subdivision process. If both encoder and decoder have previously agreed upon a set of subdivision rules and hierarchy traversal order, only the *details* to be added to the positions of the new vertices need to be sent. Those details can be considered to be prediction errors, because they measure the difference between the predicted

vertices, that result from the normal subdivision process, and the real ones, that lie on the target surface.

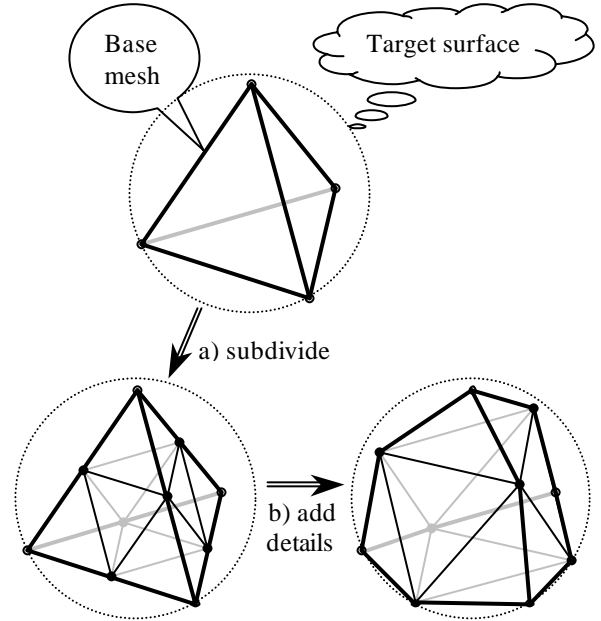


Figure 2 : Subdividing a tetrahedron with a sphere in mind.

Figure 2 illustrates this idea: after subdividing the base mesh, details are added to new vertices to have them lie on the target surface. The whole process can then be repeated using the resulting mesh as a starting point.

The more obvious advantage of transmitting these details, instead of the full vertex positions, is the one any predictive coding scheme would offer: a more compact code can be obtained by using the prediction errors rather than the predicted values themselves, since the former have a smaller variance/energy than the latter. Moreover, a smoothing subdivision scheme like the butterfly one will yield, for smooth target surfaces, smaller and smaller prediction errors in each subdivision step.

A generic predictive coding scheme would not make any use of the organisation of the set of details, which is inherently hierarchical. This can be exploited for higher coding efficiency in a way similar to that used first by Shapiro [9] with his zerotrees, and later by Said and Pearlman [6] with their SPIHT (Set Partitioning In Hierarchical Trees), to perform a fully embedded coding of the signal: an image in their cases, a 3D mesh in ours. Schröder and Sweldens [7] used a family of second-generation wavelets to represent functions on the sphere. Kolarov and Lynch [3] adapted the SPIHT algorithm to compress scalar functions defined on any 2-manifold. Their work focusses on the representation of the information distributed about a surface, but it is clear that their techniques are also applicable to the description of the surface geometry, as they suggest themselves.

3. PROPOSED TECHNIQUE

3.1. Detail hierarchy and coding algorithm

Given a finest LOD target mesh of arbitrary connectivity, we obtain through edge collapses a base mesh, which is the coarsest LOD of our pyramid (its vertices are said to be of “level/generation” 0). We subdivide it then a number of times and displace the new (level $l > 0$) vertices introduced by each step of the process to have them lie on the given mesh. If \mathbf{v} is the vertex corresponding to the mid-point of edge \mathbf{e} shared by triangles T_A (of oriented normal \mathbf{n}_A and area a_A) and T_B (\mathbf{n}_B , a_B), its associated detail \mathbf{d}_v is expressed in a local frame whose unit vectors point in the directions of $\mathbf{n} = (a_A \mathbf{n}_A + a_B \mathbf{n}_B)/(a_A + a_B)$, \mathbf{e} and $\mathbf{n} \times \mathbf{e}$.

The surface animation (or editing, which is nothing else but a form of animation) at different scales is made possible by expressing \mathbf{d}_v in a local frame tied to the coarser level mesh [10]. In that way, if a level l vertex is moved, its neighbour vertices of generation $l' > l$ follow it, so a local deformation of the surface is achieved. This kind of edit is very convenient, as it is local both in the space and frequency (level) domains, much as the one that would increase the brightness of some region of a picture.

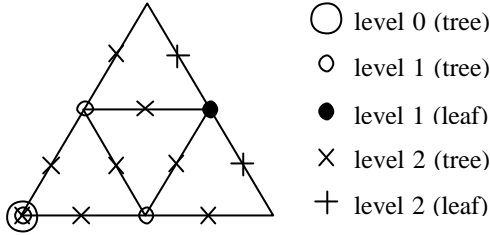


Figure 3: Hierarchical organization of details.

The hierarchical organization of the details is shown in Figure 3. For each base mesh triangle, a vertex is chosen as “responsible to cover it”. That choice also determines which detail trees will be “sterile” (“leaf” in Figure 3): those attached to the vertices that appear on the opposite edge of the base mesh triangle. In that way, any “fertile” detail tree rooted at a given vertex covers the coarsest triangle that vertex belongs to, except for the other two vertices of that triangle. The only exception to these rules has been introduced to avoid duplicating the details of vertices corresponding to the base mesh edges. The ordering of the base mesh triangles is used to determine which of the detail trees of two neighbouring triangles will cover the vertices corresponding to their shared edge.

As for the coding of details, we have followed closely the beautiful SPIHT algorithm by Said and Pearlman [6]. It consists of several steps, each corresponding to a bit-plane, and each divided into a clever sorting pass, in which some details are classified as “significant” (relative to the considered bit-plane), and a refinement pass, in which the actual bits of these significant details are output/input.

3.2. Implementation and detail file/bitstream structure

Details are uniformly quantized with a fixed number of bits (32 or 16), depending on the required resolution for details. As our experiments show clearly, the most energetic component is the one aligned with \mathbf{n} , so we assign it more bits (12 or 8) and have the other two share the rest (10 or 4 each). This quantization may seem too crude, but the visual effect on the reconstructed mesh of the least significant bits of high resolution details is completely imperceptible, in our experience.

The main problem encountered in adapting the SPIHT algorithm for 3DMC is that our details are not scalar, positive quantities belonging to a given dynamic range, but 3D vectors whose components live in presumably symmetric but otherwise unknown intervals $[-d_{max}, d_{max}]$. We have solved it by adequately interleaving the bits of the three components to form a single, scalar magnitude to be able to perform the significance test mentioned above.

The structure of the details file/bitstream which, except for the header, is fully embedded, is the following:

0. Header (25 bytes): scaling information, d_{max} for normal and tangent components, number of subdivisions and of bit-planes, N ;
1. bit-plane $N-1$: sorting bits + refinement bits;
2. bit-plane $N-2$: sorting bits + refinement bits; ...
- N . bit-plane 0: sorting bits + refinement bits.

4. RESULTS

It is difficult to compare the compression ratios we obtain to previously reported ones, as the first thing we require is that the finest LOD mesh be remeshed to have it conform to the subdivision connectivity rules. But we have tried to obtain, for each given target mesh, a base mesh with a number of triangles that would yield a remesh of approximately the same number of triangles than the original one after some number of one-to-four splits.

model	target mesh		base mesh		details	
	<i>ntri</i>	<i>size</i>	<i>ntri</i>	<i>size</i>	<i>nsd</i>	<i>size</i>
cow	5.7k	102k	802	14k	1	10.6k
dinosaur	10k	180k	607	10.7k	2	15k
fist	9.6k	173k	154	2.7k	3	13.9k
sphere	1k	18k	4	96	4	926

Table 1: Compression ratios obtained (*ntri* and *nsd* are the number of triangles and subdivisions; sizes are in bytes).

Table 1 shows some compression ratios achieved with our technique. We have adopted Li’s estimate of a raw mesh size in bytes: $size = 4 \times 3 (nvtx + ntri) \approx 18ntri$ [5]. The sizes of the base mesh and the details must certainly be added, but our base meshes are small (several hundred triangles) and could even be coded using MPEG-4v2 or Li’s tech-

niques. On the other hand, note as well that the above sizes correspond to entropy *uncoded* detail files. One can expect to obtain much higher compression ratios by feeding them into an arithmetic encoder: although it could not be used for real-time progressive coding and transmission, *gzip* reduces their size to around $\frac{1}{4}$ on the average.

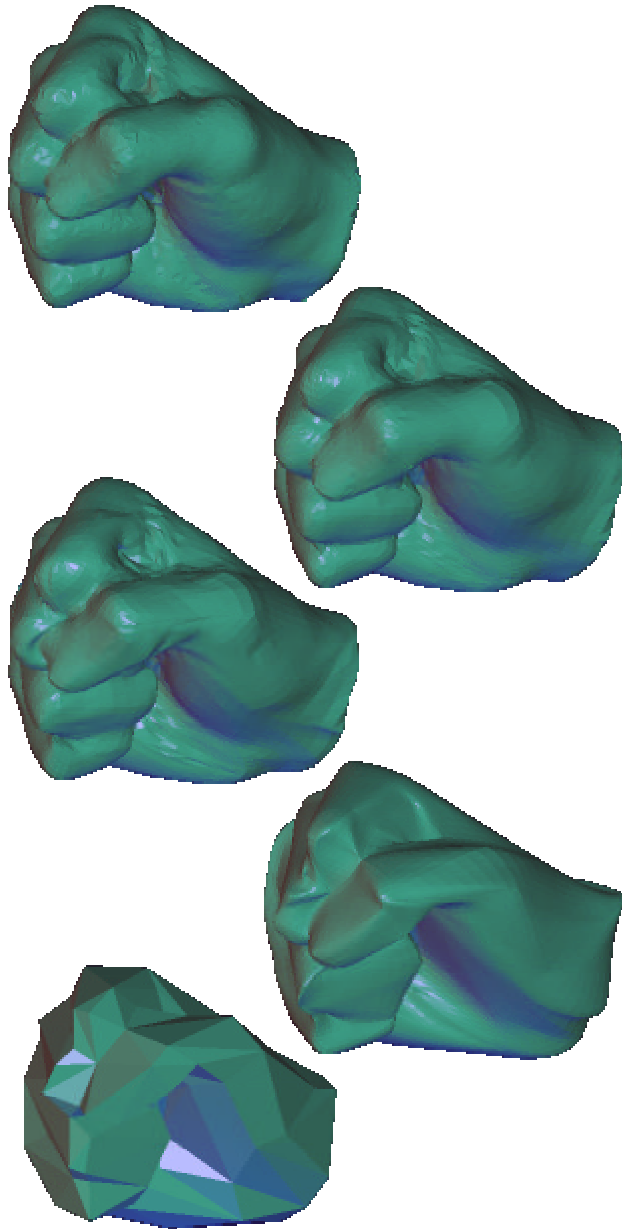


Figure 4: Five faceted renderings of a test mesh. From top to bottom: original mesh, reconstructed mesh with all, half and none of the details, and base mesh.

Figure 4 shows an original test mesh of 9.6 ktriangles, and four stages of the reconstruction of its remeshed version. Note that the base mesh can be subdivided (here, 3 times) before any details are read in order to make it smoother.

5. CONCLUSIONS AND FURTHER RESEARCH

Our technique for hierarchical coding of 3D models based on subdivision surfaces and the SPIHT algorithm yields compression ratios in the order of 10-20:1, that are similar to or better than previously reported ones. The code is fully embedded, and every bit in the file or stream may be used by the decoder to reduce the reconstruction error.

Nevertheless, the main advantage of this kind of truly hierarchical 3D object modelling, and its most important foreseeable application, is its potential to solve the problem of dynamic 3D model coding. The ISO has two versions of both VRML and MPEG-4, all of which cover exclusively the coding of static 3D meshes, with no handles for their multiresolution editing/animation. In future versions of those standards, which will have to address dynamic 3DMC, efficient means for the editing/animation of 3D objects at different scales will be needed to avoid the expensive displacement of tons of unrelated vertices.

Regarding future work that could be carried along these lines, it would be very interesting to:

- design and test some syntax for vertex displacement well suited for its efficient coding;
- allow large-scale mesh edits to be centered at arbitrary locations (and not necessarily at low level vertices);
- perform subdivision adaptively to avoid unnecessary oversampling of flat regions of the target surface.

6. REFERENCES

- [1] N. Dyn, D. Levin and J. Gregory, "A Butterfly Subdivision Scheme for Surface Interpolation with Tension Control", *ACM Trans. on Graphics*, 9(2): 160-169, 1990.
- [2] M. Eck et al., "Multiresolution Analysis of Arbitrary Meshes", *ACM SIGGRAPH Proc.*, 173-182, 1995.
- [3] K. Kolarov and W. Lynch: "Compression of Functions Defined on Surfaces of 3D Objects", *Data Compression Conference Proc.*, 281-291, 1997.
- [4] ISO/IEC JTC1/SC29/WG11, a.k.a. MPEG (Moving Picture Experts Group): "ISO/IEC 14496/AMD1", a.k.a. "MPEG-4 version 2", *ISO/IEC standard*, 1999.
- [5] J. Li and C. Kuo, "Progressive Coding of 3D Graphic Models", *Proc. of the IEEE*, 86(6): 1052-1063, 1998.
- [6] A. Said and W. Pearlman, "A New, Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees", *IEEE Trans. on Circuits and Systems for Video Technology*, 6(3): 243-250, 1996.
- [7] P. Schröder and W. Sweldens, "Spherical Wavelets: Efficiently Representing Functions on the Sphere", *ACM SIGGRAPH Proc.*, 161-172, 1995.
- [8] J. Schweitzer, "Analysis and Application of Subdivision Surfaces", *Ph.D. thesis, Univ. of Washington*, 1996.
- [9] J. Shapiro, "Embedded Image Coding Using Zerotrees of Wavelet Coefficients", *IEEE Trans. on Signal Processing*, 41(12): 3445-3462, 1993.
- [10] D. Zorin, P. Schröder and W. Sweldens, "Interactive Multiresolution Mesh Editing", *ACM SIGGRAPH Proc.*, 259-268, 1997.