Volume xx, (200y), Number z pp. 000–000

Variable Resolution 4–k Meshes: *Concepts and Applications*

Luiz Velho and Jonas Gomes

IMPA – Instituto de Matemática Pura e Aplicada,
Estrada Dona Castorina 110, Rio de Janeiro, RJ, Brazil, 22460-320.
{lvelho|jonas}@impa.br

Abstract

In this paper we introduce variable resolution 4–k meshes, a powerful structure for the representation of geometric objects at multiple levels of detail. It combines most properties of other related descriptions with several advantages, such as more flexibility and greater expressive power. The main unique feature of the 4–k mesh structure lies in its variable resolution capability, which is crucial for adaptive computation.
We also give an overview of the different methods for constructing the 4–k mesh representation, as well as the basic algorithms necessary to incorporate it in modeling and graphics applications.
Keywords: *multiresolution, four-directional grids, restricted quad-trees, multi-triangulations, adapted meshes.*

1. Introduction

Hierarchical structures are the embodiment of fundamental abstraction mechanisms that allow us to deal with complexity. For this reason, such structures are an integral part of many tools in practically every area of human activity.

Hierarchies reflect dependency relations between entities at different levels. The specific nature of these relationships is determined by the application area, and by the problem to be solved.

In Geometric Modeling and Computer Graphics, hierarchical structures are often used to represent objects with multiple levels of detail. This type of hierarchy makes it possible to process an object at different resolutions. Thus, hierarchical structures are essential for most algorithms that require adapted computations. A typical example is the visualization of 3D polygonal surfaces, where the size of polygonal facets should be proportional to the projected area on the screen.

The importance of multiple levels of detail representations has motivated the development of various hierarchical structures which, in one way or another, support that capability.

In this paper, we present the variable resolution 4–k mesh structure. It combines most properties of other multiple level of detail representations and offers several advantages over them. The 4-k mesh is a specialization of the general variable resolution structure introduced independently by De Berg et al. [1] and by Puppo et al. [2]. The simplicity of the 4-k mesh is the key to its representation power. From a theoretical point of view it assures desirable properties, such as high expressiveness, small depth and linear growth. From a practical point of view it makes the implementation simple and efficient.

The structure of this paper is as follows: Section 2 introduces some basic notions used in the other sections; Section 3 reviews general variable resolution triangulations; Section 4 discusses four directional grids and their relation to 4-k meshes; Section 5 investigates different ways to create multiresolution structures from a 4–8 tessellation; Section 6 generalizes planar four directional grids to polygonal meshes in three dimensions; Section 7 presents the variable resolution 4–k mesh that combines a four directional structure with multiresolution. Section 8 gives an overview of construction methods for 4–k meshes; Section 9 discusses level of detail operations using 4–k meshes; and finally Section 10 makes concluding remarks and points out directions for future research.

2. Basic Concepts

This section gives some definitions and basic notions that are used throughout the paper.

2.1. Meshes

A mesh is a cell complex, $K = (V, E, F)$, where V, E and F are respectively sets of vertices $v_i \in V$, edges $(v_i, v_j) \in E$, and faces $(v_i, v_j, \ldots, v_k) \in F$. The complex K provides a *topological structure* for the decomposition of two dimensional domains.

A *geometric realization* of the mesh K is created, by associating to each vertex $v_i \in V$, a coordinate value, $p(v_i) \in \mathbb{R}^n$. When $n = 2$, K is a planar mesh and when $n = 3$, K is a surface in 3D.

The 1-neighborhood $N_1(v)$ of a vertex v, consists of the set of vertices that share a face with v. The *valence* (or *degree*) of a vertex, v, is the number of edges incident in v.

A mesh can be classified according to various criteria. Here we focus on the following three: cell type; mesh structure; and mesh geometry.

According to cell type, we usually work with homogeneous meshes, where 2D cells are n-sided faces, with n constant. The most common ones are: *triangle meshes* ($n = 3$), and *quadrilateral meshes* ($n = 4$). Note that it is always possible to triangulate an n-sided face. Therefore, it is sufficient to consider only triangle meshes.

The mesh structure is related with the types of 1-neighborhoods in the mesh. In a *regular mesh*, the valence of all vertices is the same, while in an *irregular mesh* the valence may differ from vertex to vertex in an arbitrary way.

The geometry of the mesh depends on its metric properties. An *uniform mesh* is a tessellation by regular n-gons, i.e. all edges have the same size. Meshes without this property are called *non-uniform*. Note that uniform meshes are also regular.

A related geometric property is the *aspect ratio*, which measures how close a face is from a regular n-gon. We remark that there are many ways to define this quantity.

The size of a mesh, denoted by $|K|$, is the number of faces in the set F, of K.

The *resolution* of an uniform mesh is the number of vertices per unit length. The resolution of an irregular mesh can be determined locally from the length of its edges.

A mesh is called *conforming* when faces that are spatially adjacent share exactly edges and vertices on common boundaries.

Two meshes K_m and K_n are *compatible*, if there is a subset of faces $\overline{F}_m \in K_m$, that when it replaces a corresponding subset of faces in K_n, the result is a conforming mesh. Correspondence in this case means spatial overlap.

2.2. Hierarchical Structures

A *mesh hierarchy*, H, is a sequence of meshes, $H = (K_j)_{j=1,\ldots,n-1}$, such that the size of the mesh K_j increases monotonically with the index j. Furthermore, there is a dependency relation between faces at two subsequent levels j and $j + 1$, whose support overlap.

Based on these dependency relations, it is possible to construct a hierarchical structure that defines the increasing sequence of meshes H. It is also possible to define the *reverse* of the hierarchy, which is the sequence in reverse order, where the mesh size is decreasing.

A mesh hierarchy is usually constructed by local modifications that either refine or simplify some initial mesh. Thus, one can start with a coarse mesh and subdivide it by applying a refinement operator; or, alternatively, one can start with a fine mesh and coarsify it by applying a simplification operator. Figure 1 shows a scheme of this process.

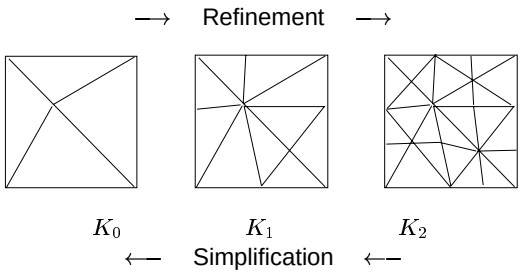


Figure 1: *Mesh hierarchy and construction mechanisms.*

Note that the modification operator provides the dependency relations necessary to build a hierarchical structure encoding the mesh hierarchy.

The nature of these operators and the method by which they are applied determines the properties of the hierarchy.

Here, we distinguish between hierarchical structures of two kinds: *non-adaptive* and *adaptive*.

Non-adaptive hierarchical structure: defines only one mesh hierarchy. Examples of this kind of structure are multiresolution and progressive meshes.

In a *multiresolution mesh*, the modifications are applied in parallel to a set of independent regions that completely cover the mesh. Each step of this process changes the mesh resolution globally. The corresponding hierarchical data structure is a tree. A multiresolution structure is usually constructed using refinement [3].

In a *progressive mesh*, the modifications are applied sequentially to only one region of the mesh at time. Each step of the process changes the mesh resolution locally. The corresponding data structure is a list. The progressive structure is usually constructed using simplification [4].

Adaptive hierarchical structure: defines a family of mesh hierarchies. One example of this kind of structure is a variable resolution mesh.

In a *variable resolution mesh*, local modifications are applied to a set of independent regions, such that, the boundary of each region remains unchanged. Note that this set of regions may not cover the mesh completely. Because of the boundary constraint, there is no interference between local modifications at each level, which can be applied independently of each other. The above property makes it possible to generate many sequences of meshes using permutations of independent local modifications. The corresponding data structure is a directed acyclic graph (DAG), that encodes dependencies across levels. A variable resolution mesh can be constructed either by refinement or simplification [5].

We remark that, in some cases, it is possible to generate an adaptive hierarchical structure from a non-adaptive structure. This conversion process usually takes advantage of the underlying multiresolution structure to compute a set of local independent modifications at each level [6, 7]

3. Variable Resolution Triangulations

This section defines more precisely some basic notions concerning adaptive hierarchical structures.

The idea of a variable resolution triangulation was introduced independently by Floriani et al. [8] and De Berg et al. [1]. Subsequently, Puppo developed an extensive theoretical framework for general variable resolution structures [2], which he called *Multi-Triangulations*. Below, we review the main concepts of this framework.

3.1. Definitions

As mentioned in Section 2, hierarchical mesh structures are based on local modifications. In the variable resolution setting, it is necessary to employ a restricted class of local modifications: the ones that are minimally compatible

A *minimally compatible local modification*, $W(K_i)$, to a sub-mesh $K_i \subset K$ of a mesh $K = (V, E, F)$, is a substitution of K_i by $W(K_i)$ in K, such that:

1. The boundary edges of K_i are not altered;
2. The interior edges of K_i are replaced by new edges.

The sub-meshes K_i and $W(K_i)$ are, respectively, the *pre-image* and *image* of the modification operator W.

Compatibility is enforced by condition (1). Since the boundary ∂K_i does not change, the new sub-mesh $W(K_i)$ is compatible with K_i and the modification operator produces a conforming mesh.

Minimality is addressed by condition (2). Since the interior of K_i changes completely, there is minimal redundancy between the sub-meshes K_i and $W(K_i)$.

The modification operator W is called *increasing* if $|W(K_i)| > |K_i|$. This means that W is a refinement operator. Similarly, W is called *decreasing* if $|W(K_i)| < |K_i|$. In this case, it is a simplification operator.

A *compatible sequence of meshes*, $(K_0, K_1, \ldots, K_n)$, is generated by the application of a sequence of modifications $(W_1, W_2, \ldots, W_{n-1})$, starting with an initial mesh K_0. This produces the sequence of meshes $(K_0, W_1(K_1), \ldots, W_{n-1}(K_{n-1}))$, where

$$K_j = W_{j-1}(W_{j-2}(\cdots W_1(K_1))) \tag{1}$$

for $j > 0$.

Note that, given an intermediate mesh, K_m, and two independent modifications W_j and W_l, that are compatible with K_m, we can apply either one of them to K_m, in order to produce a new mesh $K_{m+1} = W_j(K_j)$ or $K_{m+1} = W_l(K_l)$, with $K_j,\ K_l \subset K_m$.

The purpose of a variable resolution structure is to encode **all** possible mesh hierarchies that can be generated from a sequence of modifications $(W_i)_{i=1,\ldots,n-1}$. In order to achieve this goal, we need to distinguish between *dependent* and *independent* modifications.

A *variable resolution mesh*, $V = (K_0, W, \leq)$ is defined by an initial mesh K_0, a set of minimally compatible local modifications $W = \{W_1, W_2, \ldots, W_{n-1}\}$, and a partial order relation $\leq$ on W, satisfying the following conditions:

1. *Dependency*: $W_i \prec W_j$, if and only if there is a face $f \in F_i$ in the pre-image K_i of W_i that belongs to the image $W_j(K_j)$ of W_j. In other words, precedence is determined by compatibility of dependent modifications.
2. *Non-redundancy*: $f \in F_i$ of $W_i(K_i)$ implies that $f \notin F_j$ of $W_j(K_j)$ for all $j \neq i$. In other words, there are no duplicate faces.

3.2. Representation

The partial order relations can be described by a directed acyclic graph (DAG), where the nodes are associated with modifications W_i, and there is an arc from W_i to W_j whenever W_j is the successor of W_i according to the partial order relation $\leq$.

In order to complete the description of a variable resolution, we construct a *lattice representation* mesh by adding a source and a drain to the DAG. In this representation, each face, f is referenced by exactly two nodes. It appears in the image and in the pre-image of a modification. The node having f in its pre-image is called *successor* of f, and the node having f in its image is called *predecessor* of f.

The source node is associated with a constructor of the initial mesh K_0, and the drain node is associated with the application of all modifications W_i, $i = 1, \ldots, n-1$, to K_0, that produces the final mesh K_n. Appropriate arcs are added to and from these two special nodes.

A cut of a DAG consists of a set of nodes disconnecting it. A *front* in a lattice is a cut which contains exactly one arc for each path from the source to the drain.

Figure 14 illustrates the Lattice Representation of a 4-k mesh; and Figure 18 shows a front in this lattice.

3.3. Properties

The effectiveness of a variable resolution structure can be analyzed according to the following criteria, as discussed in [2]:

- *Expressive Power*: the number of different meshes that can be built from a variable resolution structure. It is equal to the number of distinct fronts in the lattice representation. This property is relevant to the adaptivity of the mesh;
- *Depth*: the number of levels of the longest path from source to drain in the lattice representation. This property is relevant to structure traversal operations, such as point location.
- *Growth Rate*: the ratio between the size of the longest sequence of modifications and the size of its cumulative application to a mesh. When this rate is bounded by a constant, the growth is linear. This property is relevant to the performance of selective refinement operations.

4. Four Directional Grids

This section gives some background on four directional tessellations of the plane, which are the basis of hierarchical 4-8 meshes.

4.1. $[4.8^2]$ Laves Tilings

Laves tilings are crystallographic groups that generalize regular planar tilings [9]. They are monohedral tilings with regular vertices.

In a *monohedral tiling*, all tiles are congruent to a single tile, called a *prototile*. Therefore, all tiles have the same shape and size.

A vertex, v of a tiling is called *regular* if the angle between consecutive edges incident in v is $2\pi/d$, where d is the valence of v.

Laves tilings are classified by the valences of the vertices of their prototiles.

There are eleven types of Laves tilings. Here, we focus on the $[4.8^2]$ Laves tiling, for which the prototile is an isosceles triangle with vertices of valence 4, 8, and 8. This tiling is shown in Figure 2.

This tiling has a rich fourfold set of symmetries. Also note that the $[4.8^2]$ Laves tiling is a *triangulated quadrangulation*. Thus, it combines the advantages of triangular and quadrilateral meshes.

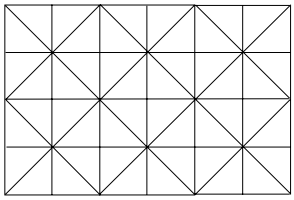


Figure 2: $[4.8^2]$ *Laves tiling*

4.2. Quincunx Lattices

The *quincunx lattice* [10] is the set of points $Q = \{Mx; x \in \mathbb{Z} \times \mathbb{Z}\}$, where M is the *quincunx matrix*.

$$M = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{2}$$

In a $[4.8^2]$ Laves tiling, we can divide the vertices into two classes: valence 4 vertices, $v \in V_4$, with $\deg(v) = 4$; and valence 8 vertices, $v \in V_8$, with $\deg(v) = 8$.

The vertices $V = V_8 \cup V_4$ of the $[4.8^2]$ tiling belong to two interleaved quincunx lattices. That is, $v \in V_8 \Rightarrow p(v) \in Q_0 = \{Mx; x \in \mathbb{Z}^2\}$, and $v \in V_4 \Rightarrow p(v) \in Q_1 = \{(1,0) + Mx; x \in \mathbb{Z}^2\}$. See Figure 3. Note that, the union of Q_0 and Q_1 is the integer lattice $\mathbb{Z} \times \mathbb{Z}$.

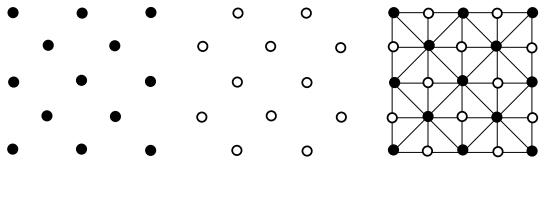


Figure 3: *Interleaved quincunx lattices and* $[4.8^2]$ *tiling*

4.3. The Four Directional Grid and Box Splines

4–8 tessellations are closely related with the four directional grid that is generated by the set of vectors $(e_1, e_2, e_1 + e_2, e_1 - e_2)$, where $e_1 = (1,0)$ and $e_2 = (0,1)$. See Figure 4.

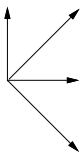


Figure 4: *Four direction vectors.*

The four directional grid is well known in the theory of

Box splines [11]. Box splines are piecewise polynomial functions created by convolution along a prescribed set of directions.

The simplest smooth box spline over a four directional grid is the Zwart-Powell element [12]. This function is piecewise quadratic, with C^1 continuity across grid lines. It is defined by the set of directions D, shown in matrix form below

$$D = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & -1 \end{pmatrix} \tag{3}$$

Note that this is the same set of vectors associated with the four directional grid.

Figure 5 shows the support of the Zwart-Powell function on the underlying grid.

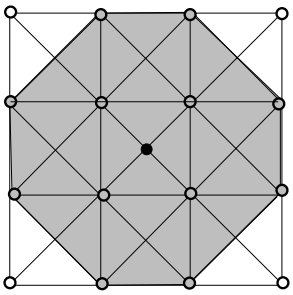


Figure 5: *Support of the Zwart-Powell function.*

5. Multiresolution 4–8 Structures

This section discusses how to define a representation for multiresolution meshes based on $[4.8^2]$ tilings.

5.1. Construction

4–8 tessellations are *refinable* tilings. This property means that it is possible to subdivide a coarse $[4.8^2]$ tiling and obtain a finer self-similar tiling. Therefore, we can construct a multiresolution 4–8 tessellation using refinement.

There are two alternative construction methods: quaternary subdivision and interleaved binary subdivision.

The *quaternary subdivision* refinement procedure of a 4-8 mesh $K = (V, E, F)$, is as follows:

1. Split all edges $e \in E$ at their midpoints m;
2. Subdivide all faces $f \in F$ into four new faces, by linking the degree 4 vertex, $v \in V_4$, to the midpoint m of the opposite edge, and also linking m to the midpoints of the two other edges.

Figure 6 shows the corresponding quaternary subdivision template.

The *interleaved binary subdivision* is as follows:

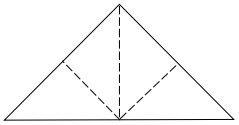


Figure 6: *Quaternary subdivision template.*

Repeat two times:

1. Split the edges $e = (v_i, v_j) \in E$ that are formed by two vertices of valence 8, $v_i, v_j \in V_8$.
2. Subdivide all faces $f \in F$ into two sub-faces, by linking the degree 4 vertex, $v \in V_4$, to the midpoint m of the opposite edge.

Figure 7 shows the corresponding interleaved binary subdivision template template.

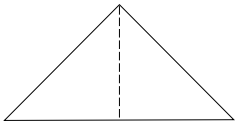
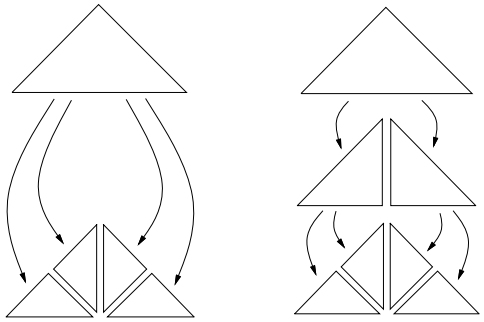


Figure 7: *Binary subdivision template.*

This recursive interleaved binary subdivision is also called red-black refinement, because at even levels “red” edges are subdivided and at odd levels “black” edges are subdivided.

5.2. Representation

A multiresolution 4–8 mesh can be represented using a tree structure of triangles. Depending on the type of refinement used, we have either a quaternary or a binary tree. See Figure 8.



(a) (b)

Figure 8: *Quaternary (a) and binary (b) tree representation of 4–8 mesh.*

These data structures are called *hierarchy of right triangles* [13, 14, 15]. Note that, in the regular case, the structure does not need to be explicitly represented.

Alternatively, we can group adjacent triangles into quadrilaterals and represent the multiresolution 4–8 mesh as a triangulated quadtree [3, 16, 7].

Note that, because of the special structure of the 4–8 tessellation, each node of the quadtree consists of four triangles created by subdivision of one diagonal of the triangulated quadrilateral. In fact, we have two interleaved quadtrees, one rotated by 45 degrees in relation to the other. This type of structure was exploited in [17]. See Figure 9.

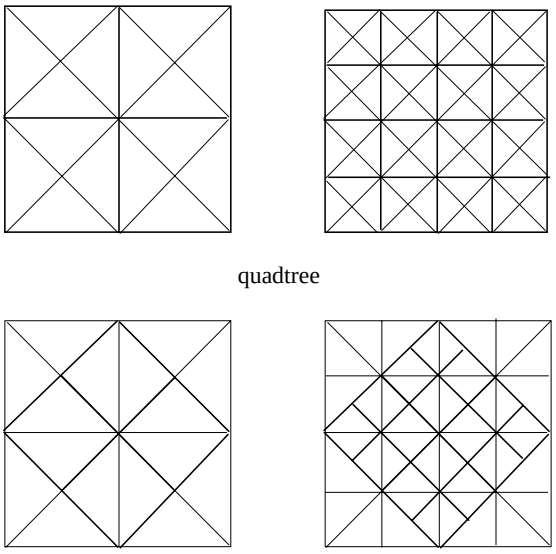


Figure 9: *Interleaved quadtrees.*

6. 4–8 Meshes

This section generalizes planar 4–8 tessellations into a family of mesh structures with similar properties. These meshes are all constructed by some form of 4–8 subdivision.

6.1. Regular 4–8 Meshes

A *regular 4–8 mesh* is a homogeneous simplicial complex that has the same connectivity of a $[4.8^2]$ tiling. All vertices in a regular 4–8 mesh have valence 4 or 8, and are called regular vertices. Moreover, the 1-neighborhood of every vertex of valence 4 has only neighbors of valence 8, and the 1-neighborhood of every vertex of valence 8 consists of a ring of vertices with alternating valences 4 and 8 (The term regular is used here in a broader sense, since the 4–8 mesh has more than one type of regular vertex).

A finer regular 4–8 mesh can be obtained from a coarse regular 4–8 mesh by refinement. (See Section 5.) Figure 10 shows a regular 4–8 mesh.

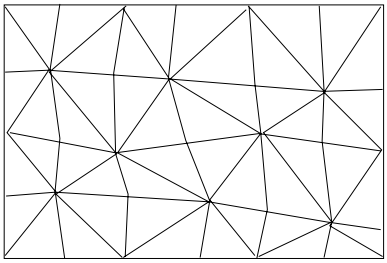


Figure 10: *Regular 4–8 mesh.*

6.2. Semi-Regular 4–8 Meshes

A *semi-regular 4–8 mesh* is a tessellation which contains isolated extraordinary vertices with valence different than 4 or 8.

This mesh structure is created from a coarse irregular mesh by applying a semi-regular 4–8 refinement method that introduces only regular vertices [18]. In that way, as the mesh is refined, extraordinary vertices from the initial mesh are surrounded by regular vertices with valence 4 or 8. Figure 11 shows a semi-regular 4–8 mesh.

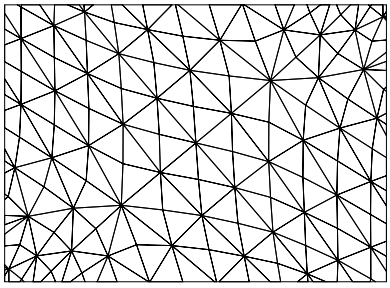


Figure 11: *Semi-Regular 4–8 mesh.*

6.3. Quasi-Regular 4–8 Meshes

A *quasi-regular 4–8 mesh* is a tessellation in which most vertices have regular valence 4 or 8, but irregular vertices are not guaranteed to be isolated. Therefore, this mesh does not possesses the 1-neighborhood structure of a regular 4–8 mesh.

This type of mesh is created by processes that almost always introduce vertices with regular valence [19].

7. Variable Resolution 4–k Meshes

This section describes a hierarchical structure to encode the family of multiresolution 4–8 meshes discussed in Section 6. This structure also allows to generate a larger class of mesh hierarchies, which we call variable resolution 4–k meshes.

A *variable resolution 4–k mesh* is a hierarchical structure that contains at each level approximately half of its vertices of valence 4 and other vertices of arbitrary valence k.

The variable resolution 4–k mesh is a special case of the variable resolution triangulation, defined in Section 3. Because of its particular nature, it has unique desirable properties not available in general hierarchical structures.

7.1. The Hierarchical 4–k Structure

The hierarchical structure of variable resolution 4–k mesh is built from a restricted set of local modifications defined on a cluster of two triangular faces. These two modifications are:

i. Internal edge split: the edge shared by two adjacent faces is subdivided, and the two faces are replaced by four faces. See Figure 12.

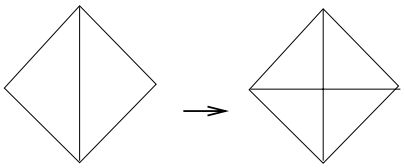


Figure 12: *Internal edge split.*

ii. Internal edge swap: The edge shared by two adjacent faces is replaced by another edge linking the opposite vertices in each face. See Figure 13

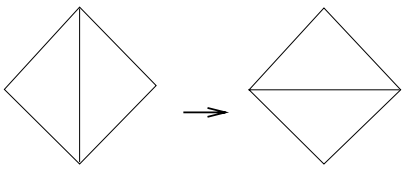


Figure 13: *Internal edge swap.*

Note that these modifications make sense only if the two-face cluster is convex. Also note that modification (i) is exactly the binary subdivision step of 4–8 refinement applied to adjacent faces. The inverse of (i) is an edge collapse. It can be shown that these operations are sufficient to make any topology preserving transformation to a mesh [20].

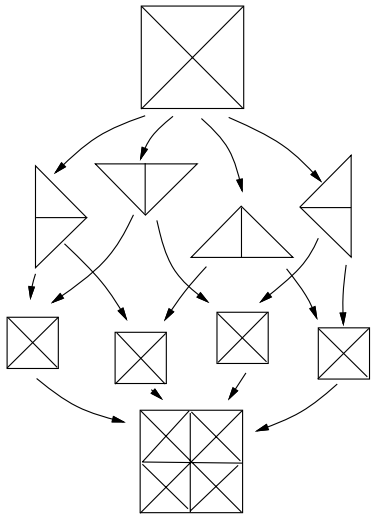


Figure 14: *Lattice representation of a variable resolution mesh (from [2]).*

Figure 14 shows an illustration of the lattice representation of a variable resolution 4-k mesh.

Another important observation is that both (i) and (ii) are edge-based modifications. We exploit this fact to design data structures for representing variable resolution meshes.

The description combines edge and face elements. Modifications of type (i) are associated with an edge that splits or collapses. The edge points out to the two faces sharing it. Additionally, a face points out to its parent and two children.

This representation is illustrated in Figure 15.

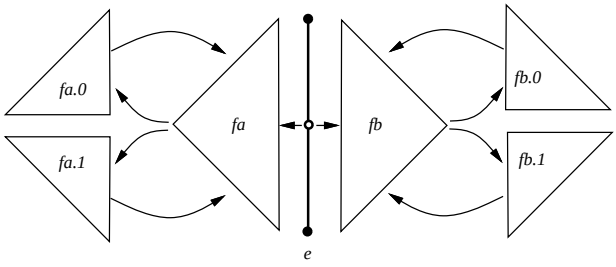


Figure 15: *Edge-face 4–8 variable resolution structures*

The specification of these data structures in pseudo-C is given below. A face is represented by the structure:

```
Face {
  Hedge*  edge[3];
  Face*   parent, children[2];
}
```

where we adopt the convention that the split edge of a face is `edge[0]` (i.e. `split_edge(f):=f.edge[0]`). When a face is bisected its edges are cyclically shifted so that `split_edge(f) = f.edge[0]`.

An edge is represented using an augmented half-edge data structure, where a pointer to the subdivided face (`fbase`) is included:

```
Hedge {
  Vertex*  point;
  Hedge*   mate;
  Face*    fbase;
}
```

These two data structures provide a compact way to encode the variable resolution 4–k mesh, as well as its inverse. They also make possible the efficient implementation of all relevant operations. For example, the pre-image of a refinement $W(e)$ is

```
Set pre_image_w(Hedge e)
{
  return make_set(e.fbase, e.mate.fbase);
}
```

The image of a refinement $W(e)$ is

```
Set image_w(Hedge e)
{
  return make_set(e.fbase.children[0],
                  e.fbase.children[1],
                  e.mate.fbase.children[0],
                  e.mate.fbase.children[1]);
}
```

The successor refinement of a face f is

```
Hedge* successor_f(Face f)
{
  return split_edge(f);
}
```

The predecessor refinement of a face f is

```
Hedge* predecessor_f(Face f)
{
  return split_edge(f.parent);
}
```

The representation of type (ii) modification, corresponding to an edge swap, it uses the same data structures. The implementation is very similar. We take advantage of the fact the each face has only one child in the context of this operation. Thus, we set `f.children[1]=NULL`.

We actually test the pointer `f.children[1]` to determine the type of modification operator.

```
int modification_type(Hedge e)
{
  if (e.fbase.children[1] == NULL &&
      e.mate.fbase.children[1] == NULL)
    return EDGE_SWAP;
  else
    return EDGE_SPLIT;
}
```

To implement the operations for type (ii) modifications the only function that needs to be changed is `image_w`. The functions `successor_f`, `predecessor_f`, and `pre_image_w` are exactly the same.

This revised implementation of `image_w` works for both type (i) and type (ii) modifications.

```
Set image_w(Hedge e)
{
  if (modification_type(e) == EDGE_SWAP)
    return make_set(e.fbase.children[0],
                    e.mate.fbase.children[1]);
  else
    return make_set(e.fbase.children[0],
                    e.fbase.children[1],
                    e.mate.fbase.children[0],
                    e.mate.fbase.children[1]);
}
```

7.2. Analysis

The DAG representation of a 4-k mesh structure may be composed by two types of nodes: a *split node* and a *swap node*, that correspond respectively to modifications of type (i) and (ii).

The split node has some special characteristics because of the nature of the refinement operator W. The number of faces in the image $W_i(K_i)$ of W_i is always 4, and the number of faces in the pre-image of K_i of W_i is always 2. As a consequence, a node W_i has exactly two incoming arcs (the two nodes that generate the faces in the pre-image of W_i), and four outgoing arcs (the four nodes that reference one of the faces in the image of W_i). This is illustrated in Figure 16.

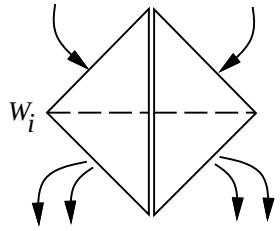


Figure 16: *Split node of the 4–8 DAG*

The split node encodes a modification that changes the resolution of the mesh. In contrast, the swap node does not change the resolution, just the local topology of the mesh. The number of faces in the pre-image of this modification operator is the same as the number of faces in the image of the operator. This number is always 2. The node has two incoming arcs and two outgoing arcs. This is illustrated in Figure 17.

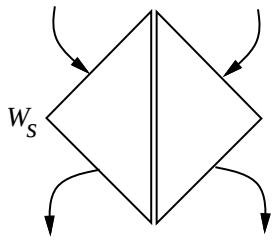


Figure 17: *Swap node of the 4–8 DAG*

The 4–k mesh has all the desirable properties, according to the three criteria used to analyze a variable resolution structure discussed in Section 3.

Below we will consider the case of a semi-regular variable resolution 4–8 mesh, which features optimal properties among the family of 4–k meshes. Subsequently, we will point out where it differs from the 4–k general case.

The semi-regular 4–8 mesh contains only split nodes. The initial mesh K_0 has arbitrary size, $|K_0| = n$. For a hierarchical structure with m levels, at each refinement step, $j = 1, m$, binary subdivision is applied to an independent set of two-face clusters that completely cover the mesh. Moreover, all clusters at subsequent levels j and $j + 1$ are interleaved. As a consequence, there are 2^j nodes in the DAG at each even level j. The size of the refined mesh produced by applying all modifications up to level j is $2^j\ n$.

The variable resolution structure of a regular 4–8 mesh has the following properties:

- *High expressive power*: It can be shown that the number, p, of distinct meshes produced by the 4–8 structure with m levels is equal to

$$p = \sum_{j=1}^{m} \sum_{k=0}^{2^j} \binom{2^j}{k} \tag{4}$$

  As an example, for $m = 6$, the expressive power is $p = 18446744078004584724$.
- *Logarithmic depth*: the number of levels of a 4–8 structure with $q = 2^m$ nodes is approximately $l = \log_2 q$.
- *Linear Growth*: the growth rate is bounded by the ratio between the sizes of the image and pre-image of the modifications, which in the case of internal edge split is 2. It can be shown that the growth rate g of a 4–8 structure is bounded by

$$g = \frac{n+2}{n+1} \tag{5}$$

Note that the semi-regular 4–8 mesh can be encoded by a pointerless structure, which implies in a compact representation.

The general 4-k mesh contains both split and swap nodes. In this case the variable resolution structure is monotonic, but not strictly monotonic. Therefore, we cannot claim the optimal properties of the semi-regular 4–8 mesh. Nonetheless, general 4–k meshes inherit most of the representation power of the 4–8 counterpart, as a consequence, they work very well in practice. Moreover, the actual properties of a variable resolution structure are largely determined by the methods employed to construct it, as will bi discussed next.

8. Construction Methods

This section gives an overview of the methods used to generate a variable resolution 4–k mesh.

We remark that it is important to have a variety of construction methods, so that they can be applied in distinct modeling situations, such as free form modeling, surface approximation, and conversion of representations, to name a few.

The main categories of methods are the ones based on refinement and simplification.

8.1. Refinement-Based Methods

We subdivide the refinement-based methods into three types: semi-regular; quasi-regular; and irregular.

The *semi-regular* refinement method employs topology based subdivision. It generalizes the regular 4–8 mesh refinement and uses interleaved edge splits. A complete description of the algorithm can be found in [18].

The method produces semi-regular meshes suitable for implementing stationary subdivision schemes. Figure 19 shows various subdivision surfaces generated with such schemes. The shape in this example is the "Stanford Bunny". The control polyhedron, shown in Figure 19(a), is a coarse mesh obtained from the original data through simplification [21].

The most natural scheme to implement using 4–8 semi-regular meshes is a generalization of subdivision for Box splines defined on four directional grids [18]. Figure 19(e) shows a C^1 subdivision surface based on the Zwart-Powell element. Figure 19(f) shows a C^4 subdivision surface based on a degree 6 Box spline.

Because of the quadrangulated structure of semi-regular 4–8 meshes, it is also suitable for the implementation of subdivision schemes originally designed for quadrilateral meshes [22, 23]. This is achieved through a decomposition of primal and dual quadrilateral refinement into interleaved binary subdivision steps [24]. Figure 19(c) shows a biquadratic B-spline surface based on the Doo-Sabin scheme. Figure 19(d) shows a bicubic B-spline surface based on the Catmull-Clark scheme.

The *quasi-regular* refinement method employs geometry sensitive subdivision. At each level, it covers the mesh with two-face clusters selected using an edge length criteria. This method produces a mesh that combines quasi-regular 4–8 topology with almost uniform geometric features. A complete description of the algorithm can be found in [19].

The quasi-regular mesh structure allows the implementation of quasi-stationary subdivision schemes. Figure 19(d) shows an example of a quasi 4–8 subdivision surface.

The *irregular* refinement method employs adaptive subdivision. It is based on multiresolution edge sampling. This method produces hierarchical meshes that conform to the shape of existing objects. A complete description of the algorithm can be found in [25].

The irregular 4–8 mesh structure is suitable to adaptive surface tessellation. Because the subdivision algorithm is very general, it can work with both parametric or implicit surface descriptions.

Figure 20 gives some examples of surfaces approximated by adapted irregular 4–8 meshes.

Figures 20(a) and (b) show a torus, defined implicitly by

$$f(x,y,z) = (x^2 + y^2 + z^2 - r^2 - 1)^2 - 4r^2(1 - z^2)$$
$$x, y \in [-3, 3], \quad z \in [-1, 1], \quad r = 1.6.$$

In Figure 20(a), we have an orthogonal projection of the base mesh together with the 3D grid; and Figure 20(b), the polygonal approximation, which contains 1324 triangles. The base mesh was constructed using a Coxeter-Freudenthal decomposition on a $4 \times 4 \times 2$ grid.

Figures 20(c) and (d), show the same torus, defined parametrically by

$$x = \cos u(r + \cos v), \quad y = \sin u(r + \cos v), \quad z = \sin v$$
$$u, v \in [0, 2\pi], \quad r = 1.6.$$

In Figure 20(c) we have the adapted decomposition of the parameter domain; and in Figure 20(d), the polygonal approximation, which contains 516 triangles. The base mesh was simply the subdivision of the rectangle $[0, 2\pi] \times [0, 2\pi]$ along its diagonal into two triangles. The algorithm has structured the parameter domain into a 4–8 hierarchy with three layers.

Note that the algorithm produces consistent results using either the parametric or implicit description of a surface.

Figures 20(e) and (f), show a digitized bust of Spock. The Cyberware data had 87040 points, structured into a regular cylindrical grid. In Figure 20(f), we have an adaptive mesh which approximates the surface within a prescribed tolerance, and in Figure 20(e), we have the corresponding domain decomposition. The facial details are clearly visible, because the regions of high curvature are sampled more densely than the rest of the surface.

8.2. Simplification-Based Methods

Simplification-based methods construct the reverse of an increasing variable resolution 4-k mesh. They start with a fine mesh and coarsify it using the inverse of an edge split operation – an edge collapse. Therefore, they produce a decreasing hierarchical structure. For several reasons, it is advisable to establish the convention that the canonical lattice representation is an increasing structure, in which the source is a coarse mesh and the drain is a fine mesh. In this context, a simplification method builds the variable resolution representation "bottom-up".

In order to perform the simplification of a mesh with regular 4–8 connectivity, it is sufficient to apply the internal edge collapse operator that transforms a cluster of four faces into a cluster of two faces (see [21]). Moreover, the simplification procedure has to ensure that clusters at subsequent levels are interleaved. Unfortunately, this type of method is only practical for regular 4–8 meshes.

In the case of arbitrary meshes, it is necessary to use also the edge swap operator. The reason is that, since an irregular mesh does not have 4–8 connectivity, it may not be possible to cover the mesh with clusters of four faces sharing a degree 4 vertex $v \in V_4$. The edge swap operator is used to modify the mesh at each level, producing the required set of four-face clusters that cover most of the mesh [21].

Figure 21 shows an example of 4–k simplification. It is a cow model distributed with SGI's powerflip demo. The initial mesh, shown in Figure 21(a) contains 5800 triangles. The sequence of simplified meshes at levels 3 to 7, is shown in Figures 21(b) through (f). They contain respectively, 1200, 700, 400, 300, and 200 faces.

9. Level of Detail Operations

This section considers the application of variable resolution 4-k meshes for managing level of detail of large geometric models. It defines the relevant operations and gives some examples. The concept of level of detail operations used in this section was developed by Floriani and Puppo [26].

9.1. Variable Resolution Queries

A level of detail operation consists in extracting a mesh K from a variable resolution structure $V = (K_0, W, \leq)$. As we have seen in section 3, this mesh $K \subset V$, corresponds to a front in the lattice representation of V, i.e., a set of arcs containing exactly one arc for each path from the source to the drain.

The collection of all nodes which can be reached from the source without traversing the arcs of the front, correspond to modifications to the mesh that are consistent with the partial order $\leq$ and produce the extracted mesh K. See Figure 18.

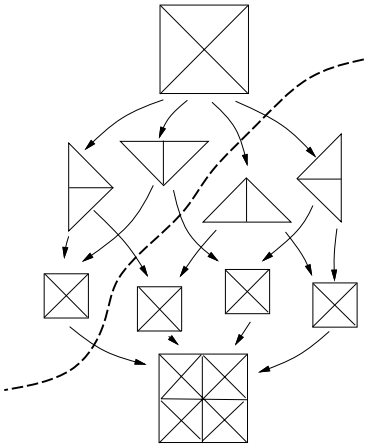


Figure 18: *A front in the lattice representation.*

We can abstract this level of detail operation as a *geometric query*, Q, to the variable resolution structure V.

This general query operation can be specified by the following parameters [26].

- An *adaptation function*: $\tau : K_j \rightarrow \{0, 1\}$, that computes some measure over V to determine if a face f produced by a modification should be accepted or not.
- A *focus set*: $S \subseteq \mathbb{R}^3$, that defines a region of interest where $\tau(f)$ is evaluated.

The answer to the query K, is the smallest conforming mesh such that $\tau(f) = 1$ and $f \cap F \neq \emptyset$, for all $f \in K_j$.

We remark that K could be either a mesh representing the whole surface, or a sub-mesh containing just the elements inside the region of interest. In the first case, the query is called *globally defined* and in the second case, *locally defined* [26].

9.2. Adapted Mesh Extraction

Examples of variable resolution query operations are: point location; region intersection; neighbor search and adapted mesh extraction. The last one is particularly important, because it appears in many graphics applications, such as, progressive rendering, real-time visualization and interactive modeling.

Adaptive mesh extraction is implemented through a selective mesh refinement procedure using the variable resolution structure [26, 27].

This procedure can use a non-incremental or an incremental algorithm. The *non-incremental algorithm* is a specialization to 4-k meshes of the algorithms described in [2] for general variable resolution structures. It starts with an initial front that contains all the arcs leaving the source node, and gradually advances the front, in a top-down fashion, based on the evaluation of the adaptation function and intersection with the focus set.

The *incremental algorithm* uses an existing front, and updates it, moving the front up or down if necessary, according to the adaptation function.

We remark that the variable resolution structure guarantees that an extracted mesh is conforming *by construction*.

Another nice feature of this framework is that, the mesh extraction procedure is independent of the query specification. As a consequence, it is straightforward to incorporate it in completely different application domains. This gives a lot of flexibility from the system design point of view.

In that context, what distinguishes two adapted mesh extraction operations is the nature of the adaptation function. Some common types of applications are related to: shape approximation; view dependent geometry, etc.

The practical performance of level of detail operations is highly influenced by the properties of the underlying structure, as noted by Puppo [2]. This is significant in the case of 4–k meshes.

Next, we demonstrate the capabilities of the 4–k mesh structure in the context variable resolution queries.

Figure 22 exhibits few examples of adapted mesh extraction, using a variety of adaptation functions, as well as, variable resolution meshes constructed using different methods.

Figures 22(a) and (b) show two meshes representing a "saddle" surface that was defined parametrically by

$$x = u, \quad y = v, \quad z = (uv)^3$$
$$u \in [0, 1], \quad v \in [0, 1].$$

The variable resolution structure was constructed using adaptive refinement. The adaptation criteria used in Figure 22(a) was triangle size. In Figure 22(b) the criteria was intersection with a rectangular region in the parametric domain.

Figures 22(c) and (d) show two versions of the "Stanford Bunny". The one in Figure 22(c) was constructed using the C^4 Box-spline subdivision scheme; and the one in Figure 22(d) was constructed using simplification. The adaptation criteria is the same for both models: it is a linear ramp in the horizontal direction determining triangle size.

Figures 22(e) and (f) show an example of point location using the cow model of Figure 21. In Figure 22(e) we have the complete mesh, in which the smallest triangle was picked by pointing at the screen. A detail of the area surrounding this point is shown in Figure 22(d).

We close this section with some remarks about a useful capability of the 4–k mesh structure that allows the construction of a triangle strip representation of the extracted mesh [28]. Similarly to selective refinement, this algorithm

starts with a path, defined on the coarsest mesh, and the path is refined while traversing the variable resolution structure. In particular, if the model has semi-regular 4–8 connectivity, it is possible to maintain a Hamiltonian path for all extracted meshes. See Figure 23 for examples.

10. Conclusions

This section concludes the paper with a review of the results and a discussion of future work.

10.1. Overview

A framework for variable resolution description of surfaces was presented. It is based on the hierarchical 4-k mesh structure. This representation has several desirable properties for multiresolution applications.

We described various methods for constructing the 4–k representation that contemplate most modeling situations. We also demonstrated the practical use of the 4–k representation, for the implementation of level of detail operations.

10.2. Future Work

Future work in this area includes: hierarchical parametrizations; multiresolution decomposition; mesh compression; and the development of an integrated application framework.

Acknowledgements

The figures in section 8 were generated with Geomview [29].

The authors are partially supported by research grants from the Brazilian Council for Scientific and Technological Development (CNPq) and Rio de Janeiro Research Foundation (FAPERJ).

References

1. Mark de Berg and Katrin Dobrindt. On levels of detail in terrains. In *Proc. 11th Annual ACM Symp. on Computational Geometry*, Vancouver, B.C., June 1995.
2. E. Puppo. Variable resolution triangulations. *Computational Geometry Theory and Applications*, 11(3-4):219–238, 1998.
3. H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, 1989.
4. Hugues Hoppe. Progressive meshes. In *SIGGRAPH '96 Proc.*, pages 99–108, Aug. 1996.
5. E. Puppo and R. Scopigno. Simplification, LOD and multiresolution – principles and applications, 1997. Eurographics'97 Tutorial Notes.
6. Hugues Hoppe. View-dependent refinement of progressive meshes. *Proceedings of SIGGRAPH 97*, pages 189–198, August 1997. ISBN 0-89791-896-7. Held in Los Angeles, California.
7. Renato B. Pajarola. Large scale terrain visualization using the restricted quadtree triangulation. *IEEE Visualization '98*, pages 19–26, October 1998. ISBN 0-8186-9176-X.
8. Leila De Floriani and Enrico Puppo. Constrained delaunay triangulation for multiresolution surface description. *Proceedings Ninth IEEE International Conference on Pattern Recognition*, pages 566–569, 1988. Held in Los Alamitos, California.
9. B. Grünbaum and G. Shephard. *Tilings and Patterns*. W. H. Freeman, 1987.
10. I. Daubechies. *Ten Lectures on Wavelets*. Number 61 in CBMS-NSF Series in Applied Mathematics. SIAM, 1992.
11. C. de Boor, D. Hollig, and S. Riemenschneider. *Box Splines*. Springer-Verlag, New York, NY, 1994.
12. P. B. Zwart. Multivariate splines with nondegenerate partitions. *SIAM J. Numer. Anal.*, 10(4):665–673, 1973.
13. W. Evans, D. Kirkpatrick, and G. Townsend. Right triangular irregular networks. Technical Report 97-0, University of Arizona, 1997.
14. P. Lindstrom, D. Koller, W. Ribarsky, L. F. Hughes, N. Faust, and G. Turner. Real-Time, continuous level of detail rendering of height fields. In *SIGGRAPH 96 Conference Proceedings*, pages 109–118, 1996.
15. M. Duchaineau, M. Wolinsky, D. Sigeti, M. Miller, C. Aldrich, and M. Mineev-Weinstein. Roaming terrain: Real-time optimally adapting meshes. *IEEE Visualization '97*, pages 81–88, November 1997.
16. Brian Von Herzen and Alan H. Barr. Accurate triangulations of deformed, intersecting surfaces. *Computer Graphics (Proceedings of SIGGRAPH 87)*, 21(4):103–110, July 1987. Held in Anaheim, California.
17. D. J. Hebert. Cyclic interlaced quadtree algorithms for quincunx multiresolution. *Journal of Algorithms*, 27(1):97–128, April 1998.
18. Luiz Velho and Jonas Gomes. Semi-regular 4–8 refinement and box spline surfaces. preprint, 1999.
19. Luiz Velho and Jonas Gomes. Quasi 4-8 subdivision surfaces. In *Proceedings of SIBGRAPI'99 - XII Brazilian Symposium on Computer Graphics and Image Processing*, Campinas, SP, Brazil, October 1999. SBC - Sociedade Brasileira de Computacao, IEEE Press.
20. Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Mesh optimization. In *SIGGRAPH '93 Proc.*, pages 19–26, Aug. 1993.

21. Luiz Velho and Jonas Gomes. Four-face cluster simplification. preprint, 1999.

22. D. Doo and M. Sabin. Behaviour of recursive division surfaces near extraordinary points. *Comput. Aided Design*, 10:356–360, 1978.

23. E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Comput. Aided Design*, 10:350–365, 1978.

24. Luiz Velho and Jonas Gomes. Decomposing quadrilateral subdivision rules into binary 4–8 refinement steps. preprint, 1999.

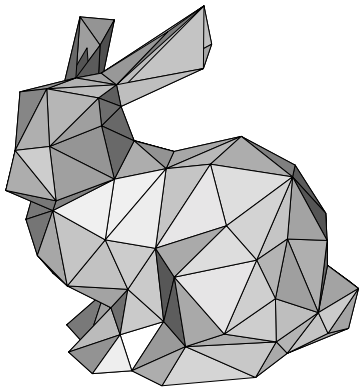
25. Luiz Velho, Luiz Henrique de Figueiredo, and Jonas Gomes. A unified approach for hierarchical adaptive tesselation of surfaces. *ACM Transactions on Graphics*, 2000.

26. Leila De Floriani, Paola Magillo, and Enrico Puppo. Efficient implementation of multi-triangulations. *IEEE Visualization '98*, pages 43–50, October 1998. ISBN 0-8186-9176-X.

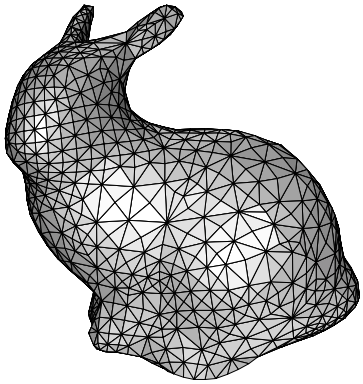
27. Luiz Velho and Jonas Gomes. Level of detail operations using 4–k meshes. in preparation, 1999.

28. Luiz Velho, Luiz Henrique de Figueiredo, and Jonas Gomes. Hierarchical generalized triangle strips. *The Visual Computer*, 15(1):21–35, 1999.

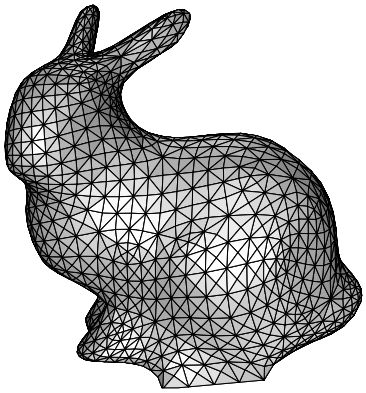
29. S. Levy, T. Munzner, and M. Phillips. Geomview. Geometry Center, University of Minnesota, 1991.



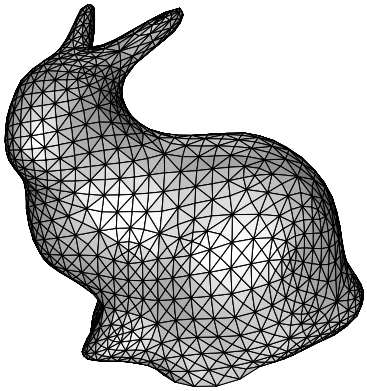
(a) Base Mesh



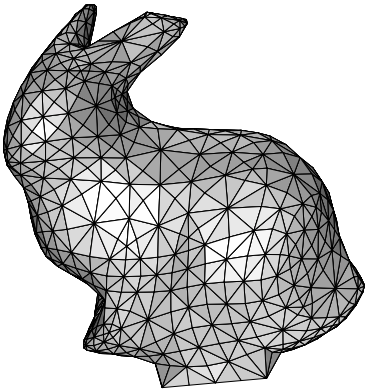
(b) Quasi 4-8



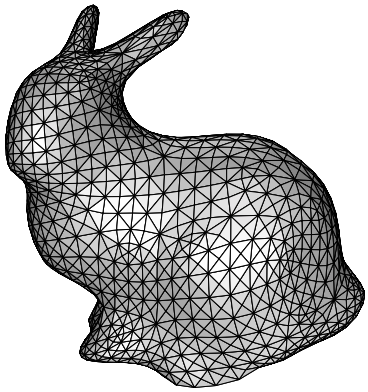
(c) Doo-Sabin



(d) Catmull-Clarck

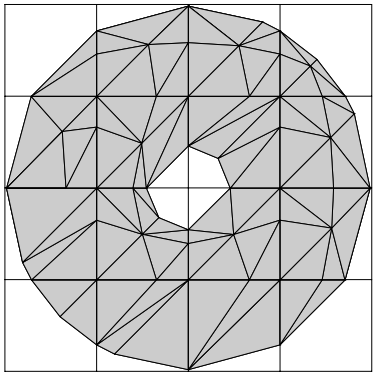


(e) Zwart-Powell

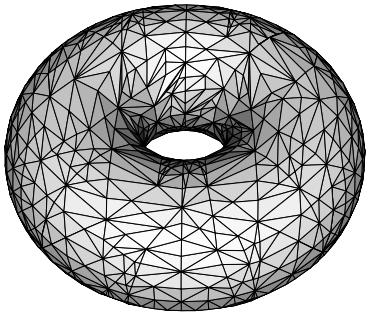


(f) C^4 Box Spline

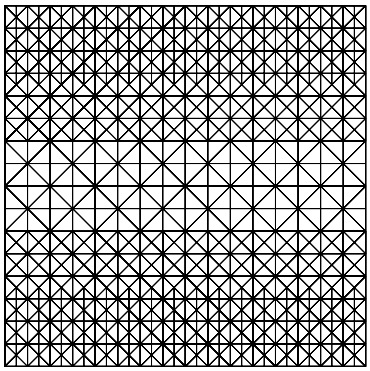
Figure 19: *Surfaces generated by different subdivision schemes based on quasi 4–8 refinement (b), and semi-regular 4–8 refinement (c-f)*



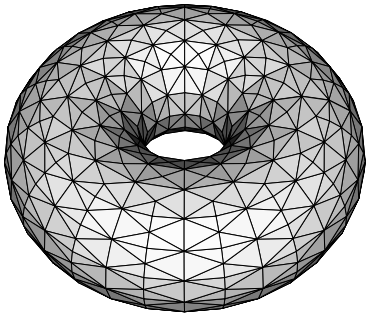
(a) spatial decomposition



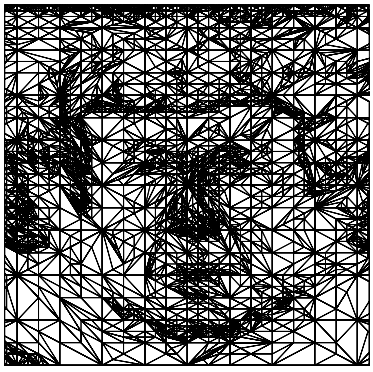
(b) implicit torus



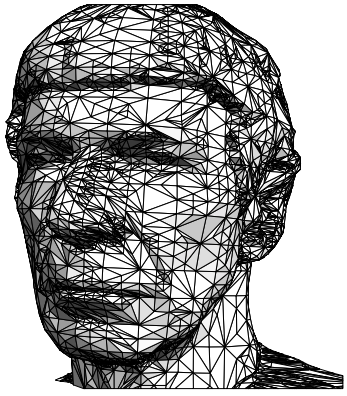
(c) UV domain



(d) parametric torus

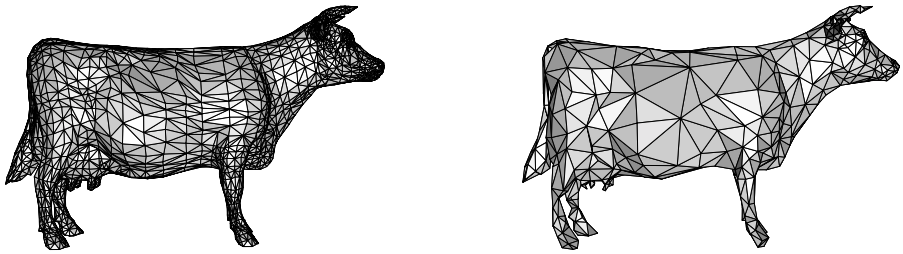


(e) UV domain

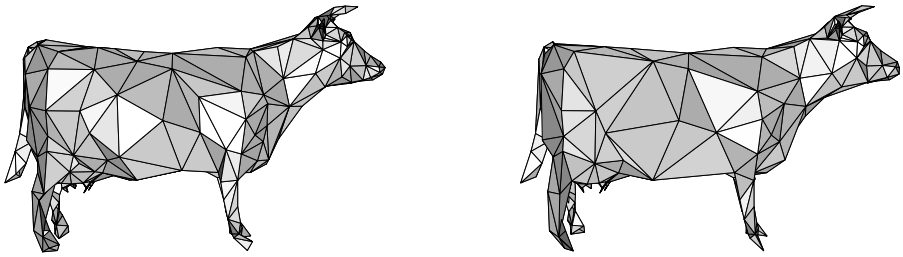


(f) Spock head

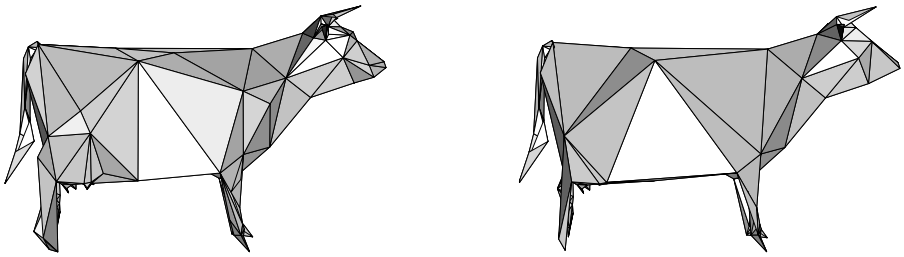
Figure 20: *Surface Approximations using adaptive 4–8 refinement of implicit (a-b), parametric (c-d), and sampled (e-f) models.*



(a) 5800 faces (b) 1200 faces

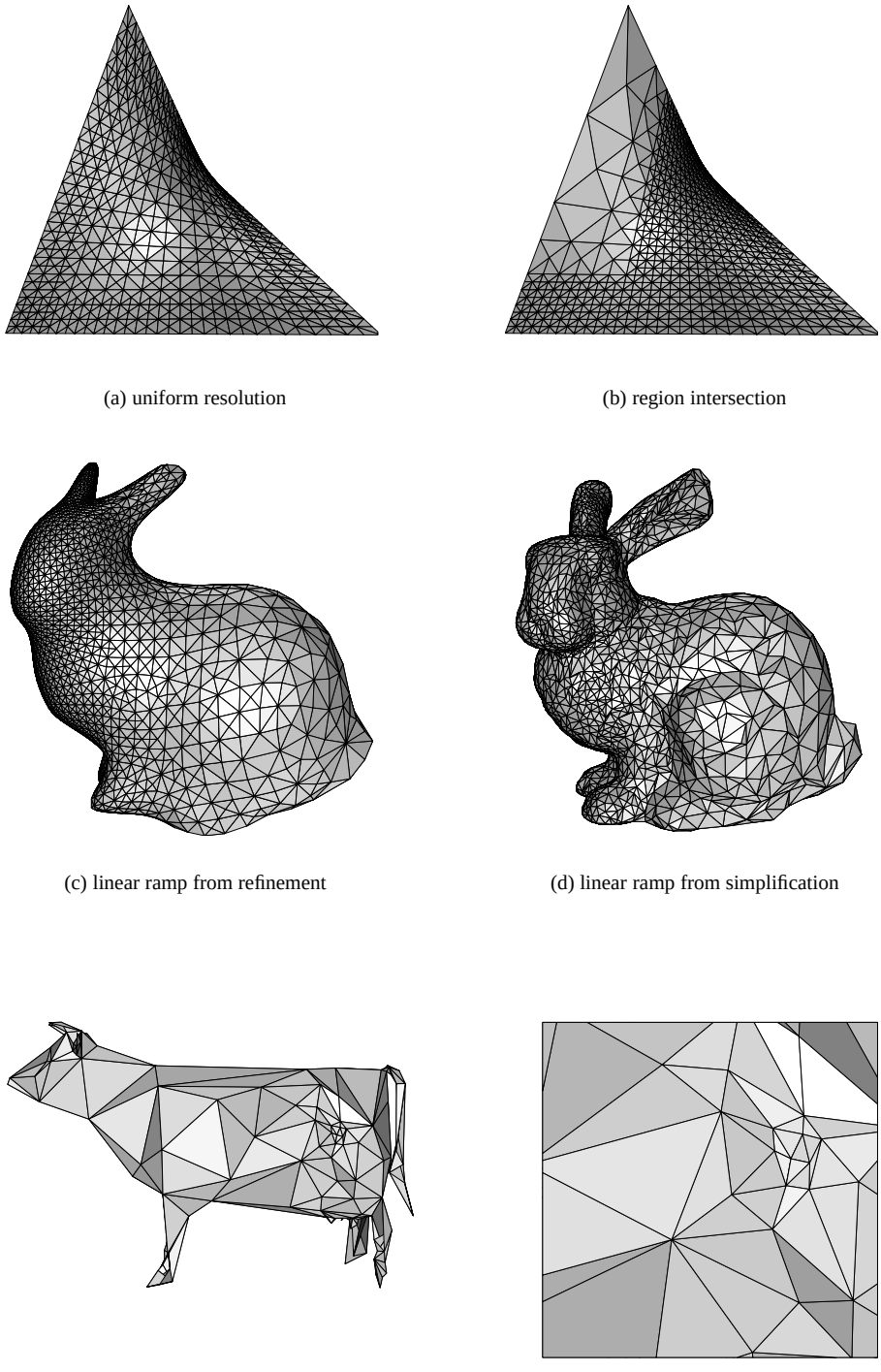


(c) 700 faces (d) 400 faces



(e) 300 faces (f) 200 faces

Figure 21: *4-k Simplification of a cow model.*



(a) uniform resolution

(b) region intersection

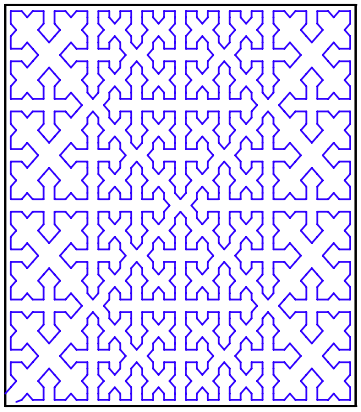
(c) linear ramp from refinement

(d) linear ramp from simplification

(e) point location

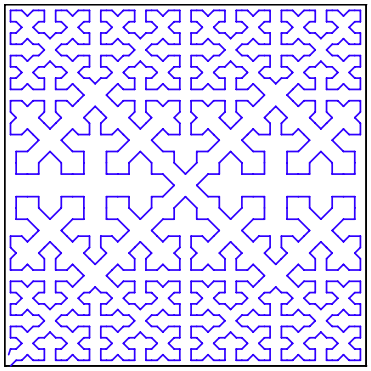
(f) detail of (e)

Figure 22: *Adapted variable resolution 4-k mesh extraction.*



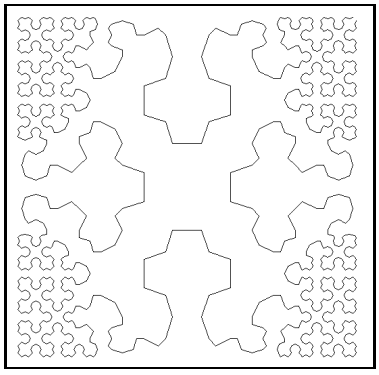
(a) uv domain

(b) sphere

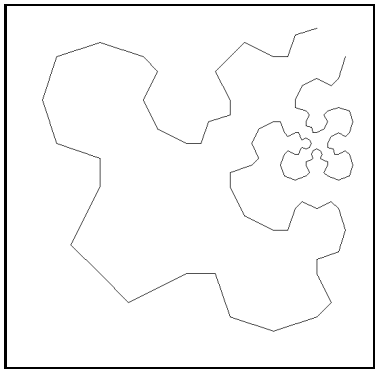


(c) uv domain

(d) torus



(e) uv domain – torus



(f) uv domain – torus

Figure 23: *Hamiltonian paths on semi-regular 4–8 meshes.*