Multiresolution Triangle Strips

Ó. Belmonte[1], I. Remolar[1], J. Ribelles[1], M. Chover[1], C. Rebollo[1], M. Fernández[2]
[1]Departamento de Lenguajes y Sistemas Informáticos. Universitat Jaume I. Castellón. Spain.
{belfern, remolar, ribelles, chover}@uji.es
[2]Departamento de Informática. Universitat de Valencia. Spain.
marcos@robotica.uv.es

ABSTRACT

Most of the previous multiresolution models use exclusively the triangle graphic primitive both in the data structure and in the rendering stage. Only a few models use another graphic primitive in this rendering process and just one uses the triangle fan as the basic primitive of the model. In this paper we present the first multiresolution model, Multiresolution Triangle Strips, that uses the triangle strip in the data structure and in the rendering stage. Each triangle strip is represented as a graph and all levels-of-detail of this strip are stored in it. The extraction algorithm traverses the graph to obtain the triangle strip at the demanded resolution. The use of this primitive speeds up the rendering process as the number of vertices sent to the graphic system is reduced.

KEYWORDS. Multiresolution, triangle strip, real time rendering, computer graphics.

1 Introduction

In most fields within Computer Graphics such as real-time rendering, virtual reality, computer games, etc., the visu-alisation process is done by rendering triangle meshes. These meshes are formed by a collection of triangles where each pair of triangles do not intersect or, if they do, it is through an edge of a common vertex. This represen-tation is quite popular due to the simplicity of the render-ing algorithm, easily implemented in hardware, and to the fact that any given polygon with a number of edges greater than three can be decomposed as a group of trian-gles.

These days Computer Graphics tends to manage polygonal models with a large number of triangles. This is due to the fact that the current data sources such as from 3D-scanners, satellite images and medical images, have a high degree of precision. If we did not use the topological mesh information there would be redundant information sent to the graphic system. Some representations, like triangle fans and triangle strips, use this topological information in order to reduce the amount of data sent to the graphics system. Most graphics libraries include these representations as rendering graphic primitives.

Depending on the application, it is often unnecessary to render all triangles; it is enough to render a simplified version of the original model. This new simplified version is termed a level of detail (LOD) of the object. Multiresolution modelling tries to adjust the level of detail depending on criteria based on the specific application and hardware.

With the exception of MOM-FAN [1], all multiresolution models use the triangle as graphic primitive representation. Hoppe [2] uses strips of triangles in the rendering stage but does not use them in the basic data structure. El-Sana [3] uses triangle strips in the data structure but the base of the model is the single triangle. First of all, the triangles not pertaining to the demanded level of detail are determined, and after that, they adapt the triangle strips. MOM-FAN [1] use triangle fans, but this does not reduce time at the rendering stage, it reduces only the storage cost of the model.

The main contribution of this paper is a new multiresolution model which uses the triangle strip graphic primitive as the base of the model. No other multiresolution model previously presented uses this primitive. In this way the amount of information sent to the graphic system necessary to visualise a certain level of detail is reduced.

Our model, Multiresolution Triangle Strips, is built from the strips of triangles found over the original polygonal model, that with highest resolution. A graph is created for each triangle strip with the information obtained after model simplification. This graph is traversed in order to recover the triangle strips at the demanded resolution.

In section 2, we discuss the triangle strip properties used in the construction of our model. Algorithms to find strips of triangles, the simplification method by Garland and Heckbert [4] and other multiresolution models using the triangle fans or triangle strip primitives, are also discussed in this section. In section 3, we show how to construct the graph used to represent the Multiresolution Triangle Strips model. In section 4, the data structure and the algorithm used to recover a demanded level of detail is shown. In the remainder of the paper, the results, conclusions and future work are presented.

2 Related work

This section begins with an overview of the triangle strip properties. The algorithm STRIPE [5] is also reviewed, as it is used to find the strips of triangles over the polygonal mesh as a first step in the construction of the multiresolution model. Many studies have been presented which dealing with the automatic simplification of polygonal models [6]. We briefly review the method based on iterative vertex pair contraction by Garland and Heckbert [4]. Finally, we review the multiresolution models using the triangle strip primitive in some stage of the rendering process.

2.1 The triangle strip primitive

As mentioned in the previous section, representing surfaces using graphic primitives such as fans or strips of triangles, reduces visualisation time and provides important memory savings. Using triangles strips, special cases exist where the adjacency of the triangles does not permit us to directly follow this encoding scheme. There we must repeat some vertices, or use the inversion operation, termed *swap*.

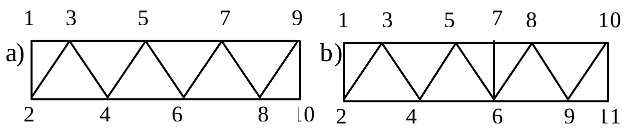


Fig. 1. Strip without swaps a), the sequence of vertices is 1-2-3-4-5-6-7-8-9-10. Strip with a swap b), the sequence of vertices is 1-2-3-4-5-**6-7-6**-8-9-10-11.

Apart from the topological properties, triangle strips have another interesting properties that are used in our multiresolution model construction. A triangle strip induces an order in the vertices forming the strip. Multiresolution Triangle Strips take advantage of this order in the construction of the triangle strips. The swap operation can be simulated by repeating vertices in the strip, so it is desirable to reduce the number of swaps. Figure 1 shows two strips, one of them with a swap, and the sequence of vertices sent to the graphic system in order to visualise them.

Some of the triangle strip searching algorithms that appear in the literature are reviewed and classified in [7]. The model developed by Evans et al. [5] finds the smallest number of triangle strips in a given mesh. Specifying different triangle strip searching criteria is possible in this algorithm. As we want to avoid the swap operation, the most interesting criterion is to select the next triangle in the strip in such a way that the swap operation is not produced. Representation of triangle strips achieved in this way, will be the shortest.

Briefly, the triangle strip search algorithm by Evans et al. [5] works on models which are not completely triangulated. The objective is the same as in that of Akeley et al. [8]: to leave the smallest amount of isolated triangles. Initially the polygons which are not triangles are grouped in regions to be triangulated at a later moment using one of the three following approximations: i) completely triangulate the model at the beginning of the search; ii) consider only one of the two triangles in cases where they can divide a rectangle when arriving at it; iii) consider the two triangles. With these criteria we decide among multiple options (tie break) to continue the strip in the following manner: i) select as the next triangle the first triangle of those adjacent to the last one introduced in the strip; ii) the same criterion as in Akeley et al. [8]; iii) alternate turns to the left and right; iv) select the next one at random; v) select the option which does not produce swaps and if that is not possible, one at random. The algorithm, called STRIPE, is in the public domain and can be found at the URL http://www.cs.sunysb.edu/~stripe/.

2.2 Polygonal mesh simplification using pair vertex contraction

The goal of polygonal surface simplification is to automatically produce approximations with fewer number of polygons than in the original mesh, and as similar as possible to the original. This new mesh has a lower level of detail or resolution. Similarity measures between an original mesh and a simplified one, can be done using an appearance-based metric [9] or a geometric measure [10].

There are important surveys where several simplification methods in the literature [6, 11, 12] are classified. Simplification algorithms based on iterative contraction are of particular interest because they have been used to construct multiresolution model representations [11, 12].

The simplification method used in Multiresolution Triangle Strips is that proposed by Garland and Heckbert [4, 11]. They have developed an algorithm for simplifying surfaces based on iterative vertex-pair contraction. A pair contraction is a natural generalisation of edge contraction where the vertex pair need not be connected by an edge. A 4x4 symmetric matrix $\mathbf{Q}_i$ is associated with each vertex $\mathbf{v}_i$. The error at the vertex is defined as $\mathbf{v}^T\mathbf{Q}\mathbf{v}$, and when a pair is contracted, their matrices are added together to form the matrix for the resulting vertex. They derive these matrices to calculate the sum of squared distances of the vertex to a set of planes.

2.3 Multiresolution modelling

Garland [11] defines a multiresolution model as a model that can provide a high number of meshes representing a single object at different resolutions, and which can be used to reconstruct any one of them on demand. Multiresolution models can be divided in two main classes: *discrete*, with a discrete number of levels and the threshold parameters to control the switching between them, and *continuous*, with a wide range of levels of detail, virtually infinite. Continuous models, according to their structure, can be subdivided into *tree-like models* and *historical models* [12].

Graphics standards such as VRML or OpenInventor use discrete multiresolution representation. Making a substantial change in appearance between two consecutive frames can lead to “popping” artefacts. This effect can be mitigated using techniques as blending or morphing.

Continuous multiresolution models represent a single object with a wide range of levels of detail. The differences from discrete models, are:
By withdrawing a small number of vertices, edges or triangles from a given level of detail, we can get another mesh which represents the object with lower resolution.
A hierarchical relationship exists among the different levels of detail of an object.
Different levels of detail can coexist in a single mesh when view-dependent visualisation is used.

2.3.1 Multiresolution models using triangle fans or strip primitives

In this section, we review the small number of multiresolution models using fans or strips of triangles in the rendering process.

Hoppe [2] has utilised strips in the rendering stage inside a view dependent multiresolution model. After selecting which triangles to draw, strips of triangles are searched. . Through experimentation Hoppe concludes that the fastest triangle strip search algorithm is a greedy one. In this greedy algorithm, each of the non-drawn triangles begins a new strip which grows through its non-rendered neighbours. In order to reduce the strip fragmentation, strips are grown in a spiral clockwise manner.

El-Sana et al. [3] have developed a view dependent multiresolution model based on an edge collapsing criterion. The first step in constructing the model is to search triangle strips on it. These triangle strips are stored in a data structure called skip list [13]. Once the multiresolution model has determined which triangles to visualise, the skip list is processed. If none of its triangles has been collapsed the strip is drawn, otherwise the skip list is processed in order to eliminate the collapsed triangles.

The work presented in [1] modifies [14] using fans of triangles as its basic representation primitive. Using this primitive, the storage cost is reduced, but the behaviour of this new model regarding its visualisation time is similar to its ancestor. A short average fan length, the high percentage of degenerate triangles, and the necessity to adjust the fans to the required LOD in real-time contribute to produce overall results which do not suppose a global improvement in visualisation time.

3. Multiresolution Triangle Strips construction

As it was previously stated, we use the simplification algorithm proposed by Garland and Heckbert in the construction stage of our multiresolution model. In this algorithm it is possible to choose the new vertex position. There are three possibilities for choosing this new position: a) to choose one of the original vertices’ position, b) to choose the midpoint between the original vertices, c) to choose that position which minimises the quadric error metric. Options b) and c) introduce new vertices. Option a) does not introduce any vertex, therefore this election produces small graphs. The size of the graph is directly related to the time necessary to extract the triangle strip at the demanded resolution.

Previous to the model construction, the triangle strips over the original polygonal model are determined. Each triangle strip is represented as a graph, codified by an adjacency list. The nodes in this adjacency list are the vertices of the strip, and the arcs are the edges. After that, the adjacency graph is updated by taking the original strips and the simplification information.

3.1 Triangle strip search

The strip search is done by using Evan’s algorithm [5]. This algorithm gets its better performance more from reducing the number of swaps than from reducing the number of strips [15].

3.2 Graph construction

Multiresolution Triangle Strip construction begins with the strip at its highest resolution. The initial adjacency graph is constructed from this. In Figure 2 a triangle strip and its representing graph are shown.

After each edge collapse the sequence of vertices representing the triangle strip is processed. The vertex which disappears is changed by the new one in this process. The triangle strip can be cut into several strips and, eventually, some of them may contain no triangles. Another irregularity, that could apeare, are those triangles which have two equal vertices after the simplification, known as degenerate triangles. The new triangle strip is checked in order to remove all of these irregularities. The new sequence of vertices indicates which edges have disappeared and which ones remain after the collapse.

We will see in § 4.1 that each `RowNode` has a field, `res`, meaning the maximum level of detail at which the vertex exists. After aech edge collapse, some edges remain and other don’t. The graph must be updated in order to capture these changes. Thre possibilities exist:

- The edge remains in the current level of detail. Its field `res` is updated with the new level of detail (Fig:1.b edge 1-2).
- The edge disapears. Its field `res` is not udated, so in the reconstruction process this arc wil not be traverse at this resolution.

- A new edge is created (fig: 1. b edge 3-2). The vertex at which the new edge arrives is added as a new node in the list of the departure node with its field `res` initialised with the current resolution.

There are some special nodes in the graph labelled BEV (Begin/End Vertex), EV (End Vertex) and SEV (Strip End Vertex). Nodes BEV and EV are used as follows: if the triangle strip is cut after a simplification (Figure 2), a node BEV will be added to the adjacency list of the node in which the edge of the strip begins. This node, BEV, means that the departure vertex is the final vertex of a sub-strip and also the beginning vertex of another sub-triangle strip. This arriving vertex will be added to the list of adjacencies of node BEV. In the same way, if the triangle strip is cut after a collapse in such a way that there is not a connection between the vertices, a new EV node will be added to the list of adjacencies of the departure node. The vertex beginning the new triangle strip will also be added to the list of adjacencies of the EV node. Finally, the SEV node means that the triangle strip has already been rendered at the current resolution. As in the other nodes, the `res` field of the SEV node represents the resolution at which the strip edge is valid.

Figure 2a) shows the original triangle strip and the sequence of edge collapses. Figure 2b) shows the graph associated with the triangle strip represented as an adjacency list. There are no new vertices introduced in the simplification process of the polygonal model.

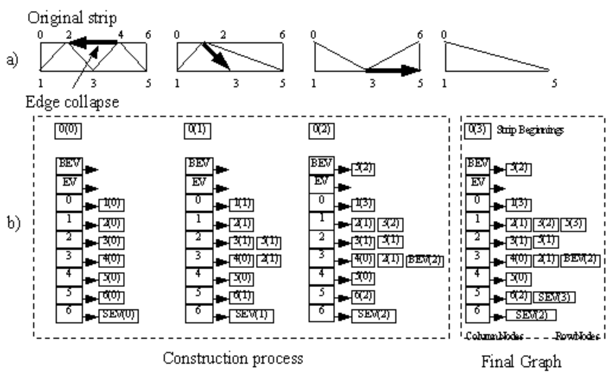


Fig. 2. a) An original triangle strip and its sequence of collapses. b) The graph updates after each collapse.

4. The model: Data structure and recovering constant level of detail algorithm

The main idea of the multiresolution model presented in this paper is to represent each triangle strip as a directed graph with weights. The graph nodes represent the vertices of the polygonal model, the directed arcs represent the edges of the polygonal model and the traverse direction represents the triangle strip sequence of reconstruction. The weights at the graph edges represent the maximum level of detail at which it is valid to traverse the edge in the reconstruction process. The reconstruction process is carried out by traversing the valid graph arcs at the demanded resolution.

This model has the important property of being independent of the simplification algorithm used in the construction process. It is also independent of the triangle strip search algorithm. For all of these we can conclude that this model is a frame of reference for all those models which will use the triangle strip graphic primitive as their basis.

4.1 Data structure

Each triangle strip is codefied as an object of the class `MultiresolutionStrip`. The list of vertices beginning the triangle strip contains information of those vertices which begin the strip and at which resolution they are valid.

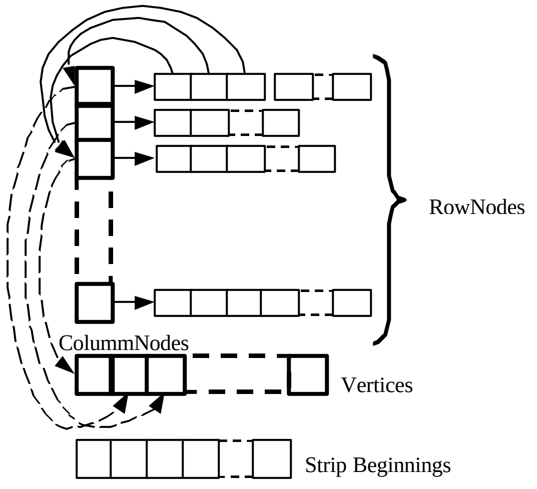


Fig. 3. The data structure representation of the multiresolution triangle strip model.

The graph is represented as a list of adjacencies. Each vertex in the triangle strip is a node graph (`class ColummNode`). Each of these nodes has a pointer to the node which represents (`vertexIndex`). The list of adjacencies contains the nodes that can be reached from it (`neighbours`), and the number of nodes in the list (`nNeighbours`). Each element in the adjacencies list of the node (`class RowNodes`) is a model edge which goes from the representant `ColummNode` to the node at `colIndex` position of the column node vector. The field `res` of each triangle strip means the maximum resolution at which it is valid to traverse the edge in the reconstruction process. Finally, the nodes in the adjacencies list are sorted by the resolution at which they appeared in the simplification process of the triangle strip. Figure 3 shows a multiresolution triangle strip represented by an adjacencies list and the data structure used. The data structure is:

```
class ColummNode {
   unsigned long vertexIndex, nNeighbours;
   RowNode * neighbours;
};
```

```
class RowNode {
   unsigned int colIndex;
   unsigned long res;
};
class MultiresolutionStrip {
   RowNode * sBegin;
   ColummNode * colVertices;
};
```

4.2 Level of detail recovery algorithm

The level of detail recovery algorithm is based on the traversal of the graph representing the multiresolution triangle strip. The traversed graph edges are those whose weights are compatible with the level of detail demanded.

Figure 4 shows two traversals of the graph in order to reconstruct the multiresolution triangle strip at the demanded resolution.

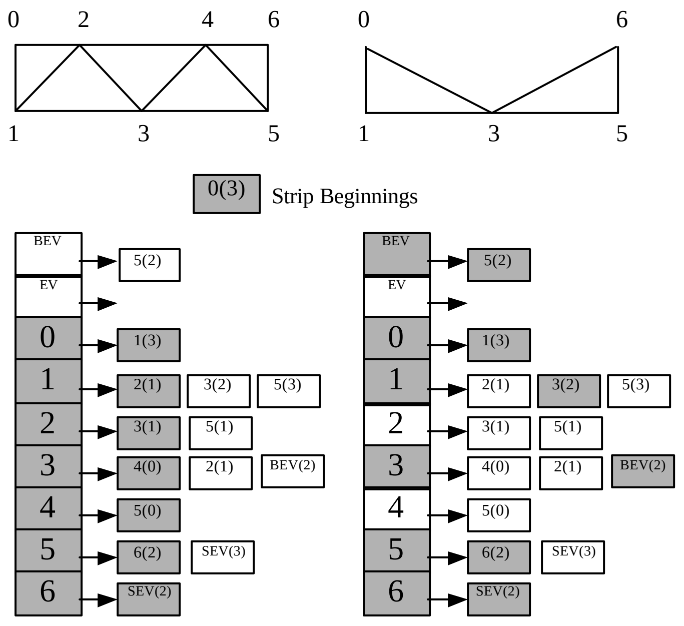


Fig. 4. Two traversals of the graph in order to extract the Multiresolution Triangle Strip at a given resolution.

```
      // First we search the strip beginning
      while BeginNotFound and NodesBeginning
1)      NextBeginning;
      end while

      if BeginNotFound exit // This strip does not
      else // exist at this resolution
        while NotEndStrip do // While there are
                                 vertices in the strip
          while Neighbour.res < ResolutionDemmand
      2a)   nextNeighbour;
          end while

2a)       if Node is not special node then
            DrawVertex;
          else if Node is BEV or Node is EV then
            NewStrip;
          else if Node is SEV then
            StripEnd = true;
          endif
        end while
      StripEnds;
      endIf.
```

List 1: Level of detail recovery algorithm.

The level of detail recovery algorithm consists of two steps: 1) The vertex with a weight higher or equal to the demanded level of detail is selected from the vertices in the beginning vertices list. This vertex is the beginning of the triangle strip. 2a) From those edges in the list of adjacencies, no traverse edge with a weight high or equal to the demanded level of detail is selected. 2b) With the node at which the edge arrives the step 2a is repeated until a SEV vertex is reached. The pseudo-code of the level of detail recovery algorithm appears in list 1.

5 Results

The experiments were carried out using a Silicon Graphics RealityEngine 2, with a MIPS R10000 at 194 MHz and 256 Mb RAM. Coding of the model was in C++ and the OpenGL graphics library was used. The characteristics of these models are shown in Table 1.

The aim of the experiments was to measure the number of vertices sent to the graphic system using Multiresolution Triangle Strips representation and compare it when using a representation based on triangles. Appendix A shows in both cases and for four different models the relation between the number of vertices sent to the graphic system and the visualised level of detail. We can observe that the curve representing the number of vertices sent to the graphic system is dramatically reduced if Multiresolution Triangle Strips is used. This reduction is higher in the first stages of the simplification, where there are more triangles to render.

Model	#Strips	#Triangles	#Vertices
Cow	136	5804	2904
Sphere	173	30624	15314
Bunny	1229	69451	34834
Phone	1747	165963	83044

Table 1: Number of triangle strips, triangles and vertices of each tested model.

6 Conclusions and Future Work

A new multiresolution model with the triangle strip as basic primitive has been presented. This model takes advantage of the triangle strip primitive and the amount of information sent to the graphic system in the rendering stage of a triangular polygonal model.

The representation of the strip as a directed graph, together with the fact that this multiresolution model is independent of the simplification algorithm used, and also independent of the searching triangle strip algorithm used, makes it a general framework for the use of the triangle strip as the basis for other multiresolution models.

Future work will address the creation of compact graphs representing the triangle strips for construction of efficient implementations of a Multiresolution Triangle

Strip representation and speed up the traversal algorithm. Retrieval of variable resolution levels of detail from a MTS representation is also another important issue. This can be easily achieved by taking advantage of vertex order in a triangle strip.

Finally, we are working to on tests to compare multiresolution models. We will use these tests to compare the current multirresolution models and also to compare our Multiresolution Strips model.

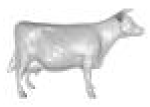
Acknowledgements

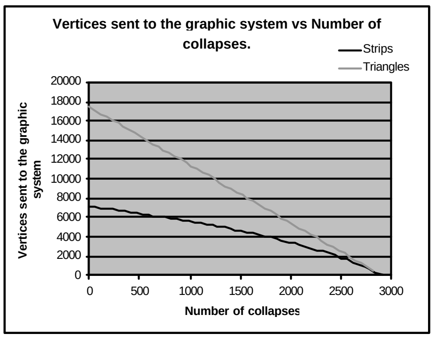
This work is partially supported by the Ministerio de Educación y Ciencia, grants TIC 1999-0510-C02-02. The presentation of this work is partially supported by Generalitat Valenciana, Conselleria d'Educació i Ciéncia.

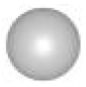
References

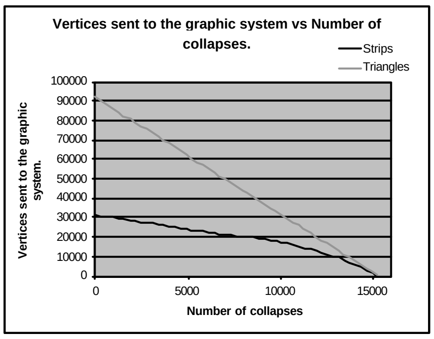
1. J. Ribelles, A. López, I. Remolar, O. Belmonte, and M. Chover. Multiresolution Modelling of Polygonal Surface Meshes Using Triangle Fans. Lecture Notes in Computer Science 1953. Proceedings of 9th Discrete Geometry for Computer Imagery Conference , pages 431-442, 2000.
2. H. Hoppe. View-dependent refinement of progressive meshes. SIGGRAPH '97, pages189-197, 1997.
3. J. El-Sana E. Azanli, and A. Varshney. Skip Strips: Maintaining Triangle Strips for View-dependent Rendering. IEEE Visualisation '99, pages131–138, 1999.
4. M. Garland, and P. Heckbert. Surface Simplification Using Quadric Error Metrics. SIGGRAPH '97, pages 209-216, 1997.
5. F. Evans, S. Skiena, and A. Varshney. Optimising Triangle Strips for Fast Rendering. IEEE Visualisation '96, pages 319-326, 1996.
6. P. Heckbert, and M. Garland. Survey of polygonal surface simplification algorithms. Multiresolution Surface Modeling Course Notes of SIGGRAPH'97, 1997.
7. O. Belmonte, J. Ribelles, I. Remolar, and M. Chover. Searching Triangle Strips Guided by Simplification Criterion. Short communications and Posters of 9-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, pages 68-75, 2001.
8. K. Akeley, P. Haeberli, and D. Burns. tomesh.c: Developer's Toolbox CD, 1990.
9. P. Lindstrom, and G. Turk. Image-driven simplification. ACM Transactions on Graphics, 19(3), July 2000, pages. 204-241. 2000
10. P. Cignoni, C. Rocchini, and R. Scopigno. Metro: measuring error on simplified surfaces. Computer Graphics Forum, 17(2): pages 167-174, June 1998.
11. M. Garland. Multiresolution Modelling: Survey & Future Opportunities. State of the Art Reports of EUROGRAPHICS '99, (1999) pages 111—131.
12. E. Puppo, and R. Scopigno. Simplification, LOD and Multiresolution - Principles and Applications. Tutorial Notes of EUROGRAPHICS'99,16(3), 1997.
13. W. Pugh. Skip lists: A probabilistic alternative to balanced trees. Communications of the ACM, Vol 33(6), pages 668-678, 1990.
14. J. Ribelles, M. Chover, J. Huerta, and R. Quirós. Multiresolution Ordered Meshes. Proceedings of 1998 IEEE Conference on Information Visualization IV '98, pages 198-204, 1998.
15. T. Möller and E. Haines. Real Time Rendering. A. K. Peters. 1999.

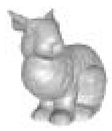
APPENDIX A: RESULTS

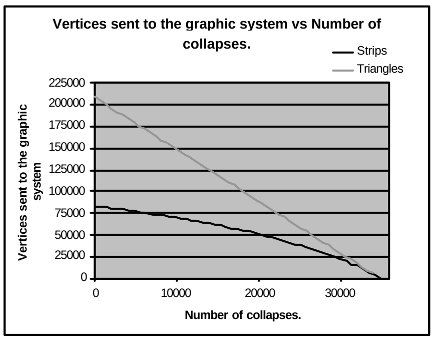


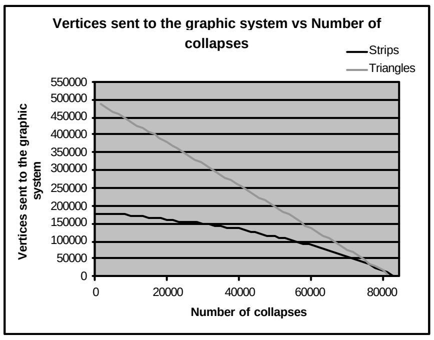

Vertices sent to the graphic system vs Number of collapses.
Strips
Triangles
Vertices sent to the graphic system
Number of collapses




Vertices sent to the graphic system vs Number of collapses.
Strips
Triangles
Vertices sent to the graphic system.
Number of collapses




Vertices sent to the graphic system vs Number of collapses.
Strips
Triangles
Vertices sent to the graphic system
Number of collapses.


Vertices sent to the graphic system vs Number of collapses
Strips
Triangles
Vertices sent to the graphic system
Number of collapses