# Triangle Strip Multiresolution Modelling Using Sorted Edges*

Ó. Belmonte Fernández, S. Aguado González, and S. Sancho Chust

Department of Computer Languages and Systems
Universitat Jaume I
12071 Castellon, Spain
oscar.belmonte@uji.es

**Abstract.** This paper presents a new multiresolution model based exclusively on the triangle strip primitive. The model is independent of the initial set of triangle strips and the simplification method used to obtain the lower resolution versions of the original model. The information stored in the data structures is sorted to minimise the time needed to recover a Level of Detail (LoD). The orientation of triangles in each strip is maintained as the model is simplified, so back-face culling can be used. The main result is a dramatic improvement in the rendering time.

## 1 Introduction

Polygonal meshes are widely used to describe surfaces in Computer Graphics, especially in Real Time Computer Graphics, and triangle meshes are the most popular representation of such models. Triangle meshes currently used in Computer Graphics are composed of thousands of triangles. A good example of this are the polygonal models used in cultural heritage [7]. Rendering these highly detailed polygonal models is a challenging problem in interactive Computer Graphics applications, where a minimum rate of 15 frames per second is needed to mimic continuous movement.

To address this problem, multiresolution models try to adapt the number of triangles in a scene according to some criterion [8, 5, 11]. The basic primitive in these models is the triangle. Only a few models use the triangle strip primitive in the rendering stage [6, 2], and only recent works have used this primitive both in the data structures and in the rendering stage [1, 9].

This work presents a continuous multiresolution model based exclusively on the triangle strip primitive. The model is independent of the initial collection of triangle strips. Unlike [9], the model is also independent of the simplification method used to obtain the lowest resolution of the original model. The inner edges of each triangle strip are sorted to achieve quick recovery of a demanded

* This work was partly supported by grant P1 1B2005-17 of Fundació Caixa Castelló-Bancaixa, and grant IST-2-004363 of the European Community.

V.N. Alexandrov et al. (Eds.): ICCS 2006, Part II, LNCS 3992, pp. 350–357, 2006.

LoD. Another characteristic that distinguishes this model from [1, 9] is that triangle orientation is maintained at any level of detail, so back-face culling can be used. The result is a dramatic improvement over models based on triangles and better performance than models based exclusively on the triangle strip primitive.

The rest of the paper is organised as follows, Section 2 includes a review of previous work Section 3 presents the new multiresolution model. Section 4 shows the experiments carried out and their results. Finally, section 5 summarises the main conclusions.

## 2 Previous Work

In [6] H. Hoppe presented a view-dependent multiresolution model in which data is stored in a hierarchy similar to that presented in [12]. After the recovery algorithm has recovered a set of triangles for a given LoD, triangle strips are searched over them. This model used triangle strips only in the rendering stage.

In [2] El-Sana et al. presented a view-dependent multiresolution model for polygonal models. The basic data structure of the model is the hierarchical tree presented in [12]. Triangle strips are searched over the model using the STRIPE algorithm [3], with the constraint of not generating strips with swap operations. These triangle strips are coded as skip-list data structures. In this case triangle strips are more closely related to the basic data structures of the model, but they are not an essential compound of it.

F. Ramos et al. [9] presented a view-dependent multiresolution model whose only data structure is the triangle strip. The building of the model used an ad hoc sequence of vertex pair contractions, so the model losses quality at lower resolutions. This model does not maintain the orientation of the triangles in the strip for every LoD, and hence back-face culling can not be applied.

In [1] Ó. Belmonte et al. presented a multiresolution model based exclusively on the triangle strip primitive. Although this model provides a frame rate that doubles those provided by multiresolution models based on triangles, its recovery time is in general higher than that offered by this latter type of model. Moreover, this model does not maintain the orientation of the triangles in the strip for every LoD, and so back-face culling can not be applied.

## 3 The Sorted Edge Triangle Strip Model

The information required to begin the building of a SETS multiresolution model is a set of triangle strips over the polygonal model. Any triangle strip search algorithm can be used to do this, for example an algorithm that takes advantage of the vertex cache size to minimise vertex cache misses or a general triangle strip search algorithm. We have chosen the STRIPE algorithm [3] because it minimises the number of swap operations in the strips.

The triangle strip search algorithm does not impose any restriction upon the simplification method that can be used to obtain the simplified version of the

original model. Furthermore, no restrictions are imposed upon the triangle strip search algorithm by the simplification method.

Any simplification method can be used on the initial set of triangle strips to obtain the simplified versions of the original model. We have opted for the QSlim algorithm [4]. This algorithm works by collapsing pairs of vertices and, given a target number of triangles, it is able to simplify the original polygonal model to reach a number of triangles near the target number. The output of this algorithm is an ordered sequence of the pairs of collapsed vertices.

If a vertex in a triangle strip collapses, the sequence that describes the strip must then be updated. A special case occurs when the vertex at the beginning of the strips changes and the previous beginning is no longer valid. Figure 1 shows a series of pairs of vertex collapses and the updated sequences of vertices.

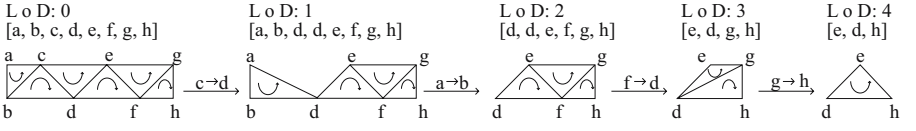


**Fig. 1.** After each pair of vertices collapses the sequence that describes the triangle strip must be updated to properly describe the new strip

For instance, let us first take vertex $b$. At LoD 0, vertex $b$ is followed by vertex $c$, which we denote by $b \rightarrow c(0)$. At LoD 1, the sequence of vertices that describes the strip has changed, and vertex $b$ is followed by vertex $d$, which we denote by $b \rightarrow d(1)$. Vertex $b$ does not appear at any other LoD.

Let us now take vertex $d$ as an example. At LoD 0, vertex $d$ is followed by vertex $e$ ($d \rightarrow e(0)$). At LoD 1, the first occurrence of vertex $d$ is followed by vertex $d$ and the second occurrence is followed by vertex $e$, which we denote as $d \rightarrow d, e(1)$. At LoD 2, vertex $d$ is followed by vertices $d$, $e$ and $g$, which is denoted as $d \rightarrow d, e, g(2)$. Finally, at LoD 3, vertex $d$ is followed by vertices $d$ and $g$ ($d \rightarrow d, g(3)$.

So, the information that needs to be store, for each level of detail, in the multiresolution model is: a) The vertex at the beginning of the strip and the maximum LoD until it is valid; b) The sorted sequence of vertices after a given one and the maximum LoD until it is valid.

### 3.1 Data Structures

A labelled graph is used to efficiently store the information needed to recover the sequence of the strip for each LoD. In this graph, each node (`ColElm`) represents a vertex in the strip, each arc (`RawElm`) is the series of vertices following on from that vertex, and the label in the arc (`RawElm.resolution`) means the maximum LoD at which the series is valid.

The vertices at the beginning of the strip (`StripBeginning`) and the maximum LoD until these become valid are stored in an array. Table 1 shows the data structure used in the model.

**Table 1.** Data structures of the model

| **StripBegenning** | **RawElm** | **ColElm** | **MultiresolutionStrip** |
|---|---|---|---|
| Integer resolution | Integer resolution | Integer idVertex | Array aStripBeginning |
| Integer idColElm | Array aIdColElm | Array aRawElm | Array aColElm |
| | | Integer shift | Array aResolution |

### 3.2 Building Process

The building process is performed just once for each polygonal model, and the resulting information is stored as the multiresolution representation of the model.

If vertex $a$ collapses over vertex $b$, all occurrences of vertex $a$ must be replaced with vertex $b$ in the sequence of vertices that describes the strip. In some cases, the sequence can be optimised by removing subsequences of the main sequence while maintaining the orientation of the triangles in the strip. The two conditions that a sub-sequence must comply with in order to be removed are: a) The number of vertices that are removed must be even, b) The remaining vertices must not add new information, that is, they must not describe triangles that do not exist before removing.

The initial sequence of vertices in Figure 1 is $[a, b, c, d, e, f, g, h]$, and the first triangle in the sequence is oriented counter-clockwise, so the triangle at the beginning of the sequence must be counter-clockwise oriented for every LoD. The first contraction moves vertex $c$ over vertex $d$ ($c \rightarrow d$) and the updated sequence is $[a, b, d, d, e, f, g, h]$; thus, no optimisation can be performed over this sequence. The next contraction is ($a \rightarrow b$), the updated sequence is $[b, b, d, d, e, f, g, h]$, and in this case it can be optimised to $[d, d, e, f, g, h]$ which is the correct counter-clockwise sequence. Finally, the third contraction ($f \rightarrow d$) yields the sequence $[d, d, e, d, g, h]$, and this can be optimised to $[e, d, g, h]$, the first triangles of which have the correct orientation.

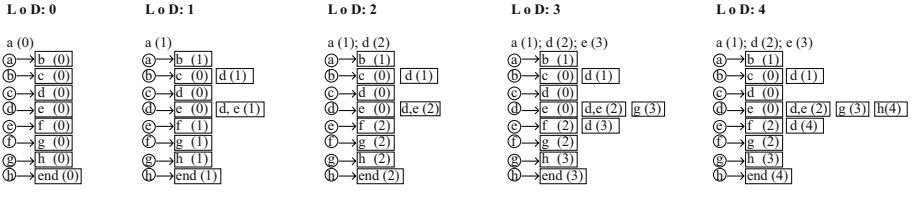


**Fig. 2.** Building process of a multiresolution triangle strip

Following with the example in Figure 1, the first LoD is 0, and for this vertex $a$ is the vertex at the beginning of the strip. This information is stored in the array of vertices at the beginning of the strip, as shown in Figure 2. Vertex $a$ has just one successor in the sequence, vertex $b$, so this is added to vertex $a$ as an arc with the current resolution. This was noted this by $a \rightarrow b(0)$. In the same way, vertex $b$ has only one successor in the sequence of vertices, vertex $c$, so this is added to $b$ as an arc with the current resolution $b \rightarrow c(0)$. The building process

continues until the last vertex in the strip, which is vertex $h$, is reached. In this case, the special arc $END$ is added to $h$, and noted by $h \rightarrow END(0)$.

At the next LoD 1, vertex $a$ is still the vertex at the beginning of the strip, so its maximum LoD must be updated. In the same way, if the successors of a vertex are the same as in the previous LoD, only the information relative to the resolution must be updated in the corresponding arc; otherwise, a new arc with the new vertices and the current resolution must be added.

### 3.3 LoD Recovery Algorithm

The recovery algorithm traverses the data structures to recover the model at the demanded LoD. The elements in the array of nodes at the beginning of the strip and the arcs for each node in the graph are sorted according to the LoD, and then a dichotomic search can be used over them. In addition, once an arc is found, this is valid for all successors of the node at the same resolution.

```
algorithm LoDrecover(res)
begin
id=aStripBeginning.dichotomicSearch(res)
while(id != END)
  storeCoords(aColElm[id].shift)
  shift = vECol[id].shift
  if shift == 0
    aColElm[id].dichotomicSearch(res)
  aColElm[id].shift++
  id = aColElm[id].aRawElm.
    current(shift)
endWhile
end
```

**Fig. 3.** Level of detail recovery algorithm

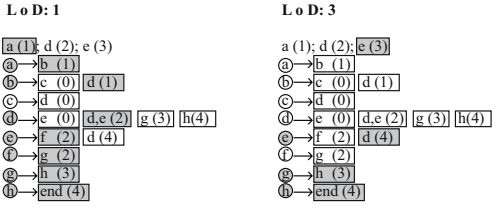


**Fig. 4.** LoD 1 is demanded on the left, LoD 3 on the right

Figure 3 summarises the recovery algorithm. The recovery algorithm starts by performing a binary search over the array of vertices at the beginning of the strip `MultiresolutionStrip.aStripBeginning` at a given resolution. If a vertex is found, its `StripBeginning.idColElm` is the identifier of the node from which the traversal of the graph must be begun. The traversal is done through arcs

whose valid resolution is strictly lower than the given resolution. If the node has not yet been visited (`ColumnElm.shift == 0`), a binary search is carried out over the arcs; otherwise, the binary search is not needed because the field `ColElm.shift` points to the next vertex in the sequence. The algorithm stops when an arc with the label $END$ is reached. Figure 4 shows two examples of the recovery process. The grayed nodes and arcs are traversed during the recovery process.

### 3.4 Coherence

Coherence means recovering only data that has changed from one LoD to another, while maintaining data that remains in both LoDs. In the SETS model this implies traversing just those graphs whose vertex sequence has changed.

To achieve this, a validity interval is defined for each strip, the bounds of which represent the minimum and maximum LoD until the current sequence is valid. This is encoded as an array containing LoDs at which a change in the sequence happens. This array is filled up when the model is loaded, so that two consecutive LoDs in the array are the bounds of the validity interval. At the beginning, the bounds in the validity interval are LoDs at position 0 and 1 of the array. When a new LoD is demanded, if it is within the validity interval the sequence that represents the strip is valid, otherwise the graph must be traversed. Finally, the bounds of the interval are updated, the maximum bound being the first element in the array greater than the current LoD, and the element in the previous position is the minimum bound.

## 4 Experimental Results

This section begins by showing the spatial cost of the multiresolution model. The, visualisation time of the SETS model is then compared with a modification of the MTS model [1] in which the orientation of the triangles in the strip is preserved as the model is simplified.

The tests were conducted using the models whose characteristics are shown in Table 2. The platform used was a Pentium IV processor at 1.8 GHz. with 256 Kb cache, 1 Gb RAM and an nVidia GeForce3 Ti 200 graphics card. The compiler used was gcc 3.3.3 version in a Linux 2.6.5 kernel.

Table 2 also shows the memory size of the SETS multiresolution models compared to those for the MTS multiresolution models. In all cases the size of the SETS models are bigger than the MTS models, but we think that the better performance of the recovery algorithm offsets this disadvantage.

**Table 2.** Polygonal data and spatial cost of the multiresolution models in Mb

| Model | #Vertices | #Triangles | #Strips | SETS | MTS |
|---|---|---|---|---|---|
| Cow | 2904 | 5804 | 136 | 0.338 | 0.253 |
| Bunny | 34834 | 69451 | 1229 | 4.150 | 2.964 |
| Phone | 83044 | 165963 | 1747 | 9.832 | 6.766 |

The tests conducted to measure the performance of the recovery algorithm were those defined in [10]. Each test recovers a percentage of the total number of LoDs, and the total time is averaged over the number of LoDs recovered. The difference between the tests is the distribution taken to recover the LoDs: linear or exponential. The linear distribution takes LoDs with the same distance in between; the exponential distribution takes close LoDs when they are next to the maximum resolution, and distant LoDs when they are next to the minimum resolution. These two tests can be interpreted as the model moves towards or away from the observer.

In both tests 1, 3, 6, 10, 15 and 20% of the total number of LoDs present in the multiresolution model were recovered. To avoid possible influences of the direction of the tests, these start at the maximum LoD, descend towards the minimum LoD and then return to maximum LoD.

Table 3 shows that the more LoDs are recovered the lower the average time is, regardless of the model and the test used. This behaviour is due to the use of coherence, the lower the distance between LoDs is, the more LoDs are recovered.

**Table 3.** Visualisation time results for the tests. Time in ms

| | **Cow** | | | | **Bunny** | | | | **Phone** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *Linear* | | *Exponential* | | *Linear* | | *Exponential* | | *Linear* | | *Exponential* | |
| *LoD* | MTS | SETS | MTS | SETS | MTS | SETS | MTS | SETS | MTS | SETS | MTS | SETS |
| 1% | 1.186 | 1.694 | 1.186 | 1.525 | 11.951 | 11.922 | 12.253 | 11.520 | 26.683 | 25.189 | 28.194 | 24.346 |
| 3% | 1.200 | 1.371 | 1.086 | 1.428 | 10.010 | 8.544 | 10.761 | 8.645 | 22.914 | 18.089 | 25.156 | 18.107 |
| 6% | 0.974 | 1.146 | 0.974 | 1.117 | 9.143 | 7.071 | 10.012 | 7.401 | 21.427 | 15.107 | 23.918 | 15.645 |
| 10% | 0.929 | 0.963 | 0.947 | 0.998 | 8.715 | 6.342 | 9.700 | 6.741 | 20.625 | 13.604 | 23.224 | 14.380 |
| 15% | 0.861 | 0.861 | 0.873 | 0.872 | 8.468 | 5.908 | 9.497 | 6.352 | 20.206 | 12.757 | 22.881 | 13.691 |
| 20% | 0.818 | 0.775 | 0.861 | 0.792 | 8.334 | 5.659 | 9.383 | 6.182 | 19.940 | 12.359 | 22.664 | 13.301 |

The visualisation time is higher for the exponential than for the linear test. This is due to the fact that the number of vertices recovered is higher in the exponential than in the linear test, and so more time is used in rendering.

The SETS model yields better results than the MTS model, as it has more triangles. This is due to the fact that data is stored in the data structures of the SETS model in a sorted way, and in this way the sequence that represents a triangle strip can be recovered faster than in the MTS model. The SETS model also yields better results as more LoD are recovered in all cases.

## 5 Conclusions

A new multiresolution model based only on the triangle strip primitive has been presented. The main characteristic of the model is that data is stored in a sorted fashion, taking into account the maximum LoD at which any inner edge exists. The building of the models is independent both of the algorithm used to find a collection of triangle strips and the simplification method use to obtain simplified

versions of the original model. In addition, the orientation of the triangles in a strip is preserved as it is simplified so that back-face culling can be used to speed up the rendering stage.

The main result is a speeding up of the visualisation time, 70% faster in the best case, as compared to that offered by models that do not use sorted data.

The main drawback is an increase in the spatial cost of the models in main memory, but we believe that this is well justified taking into account the better performance offered by the model.

## References


1. Ó. Belmonte, I. Remolar, J. Ribelles, M. Chover, M. Fernández. Efficiently Using Connectivity Information between Triangles in a Mesh for Real-Time Rendering. *Future Generation Computer Systems*, 20(8), pp: 1263-1273, 2004.
2. J. El-Sana, E. Azanli, A. Varshney, Skip Strips: Maintaining Triangle Strips for View-dependent Rendering, *IEEE Visualisation '99*, pp:131-138, 1999.
3. F. Evans, S. Skiena, A. Varshney, Optimising triangle strips for fast rendering, *IEEE Visualisation '96*, pp: 319-326, 1996.
4. M. Garland, P. Heckbert, Surface Simplification Using Quadric Error Metrics, *Proceedings of SIGGRAPH '97*, pp: 209-216, 1997.
5. M. Garland. Multiresolution modelling: survey & future opportunities, *State of the Art Reports of EUROGRAPHICS '99*, pp: 111-131, 1999.
6. H. Hoppe, View-dependent refinement of progressive meshes, *Proceedings of SIGGRAPH '97*, pp: 189-197, 1997.
7. M. Levoy et al, The digital Michelangelo project: 3D scanning of large statues. *Proceedings of SIGGRAPH 2000*, 2000.
8. D. Luebke, M. Reddy, J. D. Cohen, A. Varshney, B. Watson, R. Huebner. Level of detail for 3D graphics. *Morgan-Kaufmann*, 2003.
9. F. Ramos, M. Chover, LodStrips, *Proceedings of Computational Science ICCS 2004*, pp: 107-114, 2004.
10. J. Ribelles, M. Chover, A. López, J. Huerta, A First Step to Evaluate and Compare Multiresolution Models, *Short Papers and Demos of EUROGRAPHICS'99*, pp: 230-232, 1999.
11. J. Ribelles, A. López, O. Belmonte, I. Remolar, M. Chover. Multiresolution modeling of arbitrary polygonal surfaces: a characterization. *Computers & Graphics*, 26(3), pp: 449-462. 2002.
12. J. Xia, A. Varshney. Dynamic View-Dependent Simplification for Polygonal Models, *Visualization '96 Proceedings*, pp: 327-334, 1996.