A First Step to Evaluate and Compare Multiresolution Models

J. Ribelles, M. Chover, A. López and J. Huerta

Universitat Jaume I, 12071, Castellón, Spain

Abstract

A variety of multiresolution models for arbitrary triangle meshes have been presented in the last years. In order to achieve an interactive visualization, multiresolution models must store and retrieve data efficiently. In this paper, we present a first step to evaluate and compare multiresolution models. We propose a set of tests to measure the speed performance of the models for interactive visualization and uniform resolution LOD. We have carried out some experiments with implementations of the following two models: Progressive Meshes and Multiresolution Ordered Meshes.

1. Introduction

Interactive applications have increased enormously their use of highly detailed objects in the last few years. These objects are often represented as polygonal meshes for efficient display and the resulting meshes involve an important increase of storage and transmission costs, and make more difficult, if not impossible, interactive visualization. The use of simplification methods[1] is a solution to this problem that achieves a compact approximation of the original model with a smaller set of polygons and guaranteed approximation error, reducing disk and memory requirements and speeding network transmission. To further improve rendering performance, it is common to define various LODs of the object. The coarsest approximations are used when the viewer is far from the object, while higher detail versions are substituted as the viewer approaches. However, visual discontinuities are encountered when switching between two LODs.

Multiresolution modeling[2] tries to solve this problem by maintaining multiple LODs. It has been proposed as an efficient method for representing complex meshes and specially useful for interactive visualization, progressive transmission, geometry compression, LOD approximation and selective refinement. Consequently, a multiresolution model must define an adequate frame for, on the one hand, storing the different resolutions and, on the other hand, managing them.

In this paper, a first step towards the evaluation and comparison of multiresolution models is presented. Until now, studies with this objective have not been published. It is very difficult to make a general evaluation of models due to the different purposes for which they have been developed. For this reason, we have focused our initial work on interactive visualization with uniform resolution LOD. For this purpose, a set of tests has been defined consisting of criteria to measure the speed performance of a model in such a way that the obtained results can serve as a reference. It is also interesting to study the storage space efficiency of the model and try to establish some kind of relationship between the storage cost and speed requirements. Although there is not a specific test, the storage costs of the models are attached in the section of experimental results. For the experiments, two multiresolution models have been implemented: Progressive Meshes, introduced by Hoppe[3], and Multiresolution Ordered Meshes, introduced by Ribelles et al.[4]. The first model has been selected as an example of an implicit model (in our implementation only has been encoded the geometry[5]) and, in addition, it is an available feature of Microsoft DirectX 5.0. The other one is an explicit model. Conclusions can be reached about concrete models by the use of tests.

2. Tests

The proposed tests are designed to measure the speed performance of the evaluated model. Each test performs a number of requests of visualization with different levels of detail and they differ in the way the sequence is generated. If a model consists of n LODs, each test i generates a sequence of m_i LODs, $m_i < n$, with the following criteria.

Published by the Eurographics Association, ISSN 1017-4656.

2.1. Random test

This test generates a random sequence of LODs. The aim is to measure the performance of a model when consecutive requests consist of unrelated LODs, thus allowing sharp changes between consecutively requested LODs. The following code generates the random sequence:

srand (*seed*);
for i=1..m_r **do**
  LOD[i] = f(**rand**());
end for

where *seed* is a positive integer constant, and f is a function that normalizes the random number to the range $[1..n]$. This code generates the same random sequence given a constant *seed*. Therefore, it is possible to test different models with the same conditions. In order to make the test independent of the length of the sequence, the mean of the time, $\bar{t}$, measured from experiments should be considered. Thus, the total time, t_r, is divided by the total number of LODs tested, m_r, provided that m_r is large enough,

$$M_r = \frac{t_r}{m_r}$$

2.2. Linear test

The linear test consists of a linear sequence of LODs. The aim is to measure the performance of the model when LODs requested consecutively are related linearly. The difference between consecutive LODs will be a constant step, proportional to the total number of LODs. We must take into account two important features of this test: the increasing and decreasing order and the amount of the step between LODs that are consecutive in the sequence. The following code generates an increasing linear sequence at a given step k:

$m_l = n/k$;
for i=1..m_l **do**
  LOD[i] = $i * k$;
end for

The order of the sequence could lead to different test results, due to the fact that some models do not have the same behaviour when refining than when decimating. Therefore, two kind of tests are necessary: the increasing order test (traverse the sequence forwards) and the decreasing order test (traverse the sequence backwards). The final result will be the mean of both tests. The step value is also significant, therefore we propose a set of steps computed as a percentage of the number of LODs of the model, $k = 0.5\%n, 1\%n, 2\%n, 5\%n$, in order to obtain test sets of length m_l = $200, 100, 50$ and 20, respectively. Despite the fact that interactive applications usually require a display of 25 frames per second, the proposed percentages are designed to evaluate the time necessary to visualize the model under different conditions.

2.3. Exponential test

The exponential test consists of an exponential sequence of LODs. The aim is to measure the performance of the model when LODs requested consecutively are related exponentially. The curvature of the exponential relationship between consecutive LODs will be designed in such a way that changes between LODs of lower resolution are smaller and changes between LODs of higher resolution are greater. The length of the sequence, m_e, determines the steps between consecutive requested LODs. Analogously to the linear tests, we propose tests with both the increasing and decreasing order, and a set of values, m_e= 200, 100, 50 and 20, in order to test the models under different conditions. The following code generates an increasing exponential sequence of a given length m_e:

$x = 0$;
for i=1..m_e **do**
  LOD[i] = $f(\frac{(1+C)^x-1}{C})$;
  $x = x + \frac{1}{m_e-1}$;
end for

where C is a constant such that approximately 80% of the resulting LODs are in the set of $n/2$ LODs of lower resolution and 20% are in the set of $n/2$ LODs of higher resolution, that is, $C = 20$, and f is a linear transformation of the range [0..1] of the fraction into the index range of LODs [1..n], such that 0 corresponds to the lowest resolution LOD and 1 corresponds to the highest resolution LOD. The increasing and decreasing order tests consist of traversing the four proposed sequences forwards and backwards, respectively.

3. Results

The experiments have been performed using a Silicon Graphics workstation Reality Engine 2, with a MIPS R10000, 194MHz and 256Mb RAM. Models have been encoded in C and OpenGL, and the simplification method is the proposed by Garland and Heckbert[6].

Table 1 shows the mean retrieval time (in miliseconds) using the two implemented models: PM and MOM. For each one, the experiments have been performed with different data sets (figure 1), different number of LODs (20, 50, 100 and 200) and different sequence type (random, linear and exponential). The random test shows that MOM is more efficient when abrupt changes in the sequence of LODs are required. As the complexity of the model increases, the difference in the results becomes higher. Finally, we can observe that the random test is independent of the length of the sequence.

By comparing the random and linear tests, we can observe that the retrieval time of a LOD in MOM is independent of the previously displayed LOD. It behaves constantly with respect to the length of the sequence while PM is better as the difference between consecutive LODs is smaller. Moreover,

Model		Random		Linear		Exponential	
		Mom	Pm	Mom	Pm	Mom	Pm
	20	0.5	4.0	0.6	0.8	0.3	0.8
Cow	50	0.6	5.0	0.5	0.8	0.4	0.5
	100	0.4	5.1	0.5	0.8	0.3	0.4
	200	0.5	5.3	0.6	0.6	0.2	0.4
	20	3.0	27	3.6	6.9	1.7	4.6
Sphere	50	3.8	31	3.1	4.1	1.8	2.6
	100	3.4	31	3.2	3.6	1.7	2.3
	200	3.3	29	2.9	3.3	1.6	1.9
	20	13	80	14	26	6.7	15
Bunny	50	15	90	14	13	6.6	8.9
	100	13	85	13	11	6.8	6.7
	200	13	82	13	10	6.8	5.6
	20	34	269	37	63	18	48
Phone	50	38	313	35	41	18	28
	100	36	289	35	34	18	21
	200	35	281	35	31	18	17

Table 1: *Tests results (in miliseconds)*

for the linear test the difference between mean retrieval times of the two models is smaller than in the random test.

By comparing the linear and exponential tests, we can observe that the mean retrieval times have been reduced considerably, given that a higher number of LODs of lower resolution are retrieved. The difference is more significant in the case of MOM, where the exponential results become half the linear results. Again, the test shows that PM is more efficient when the difference between LODs is smaller.

Table 2 shows that the storage cost of the two models is very similar. Therefore, we cannot establish a relationship between storage cost and speed efficiency.

4. Conclusions

A first step to evaluate and compare multiresolution models has been presented. The proposed tests evaluate the speed performance of the models for interactive visualization and uniform resolution LOD. Experiments with two models have been performed in order to validate their significance. The tests evidence the features of each multiresolution model.

Future work will aim to improve and make more tests to measure, for example, the relationship between storage cost and speed efficiency. One important issue that remains to be studied in more detail is adapting the tests to variable resolution LODs and focusing the tests from other points of view like progressive transmission or storage compression. Finally, we intend to make a comparison including a larger set of multiresolution models, including not only geometry but appearance attributes.

Attribute	Cow	Sphere	Bunny	Phone
Vertices	2,904	15,314	34,834	83,044
Faces	5,804	30,624	69,451	165,963
Mb.	0.102	0.538	1.222	2.918
LODs	2,807	15,263	33,942	81,647
Mb PM	0.255	1.349	3.065	7.316
Mb MOM	0.251	1.364	3.028	7.285

Table 2: *Storage costs*

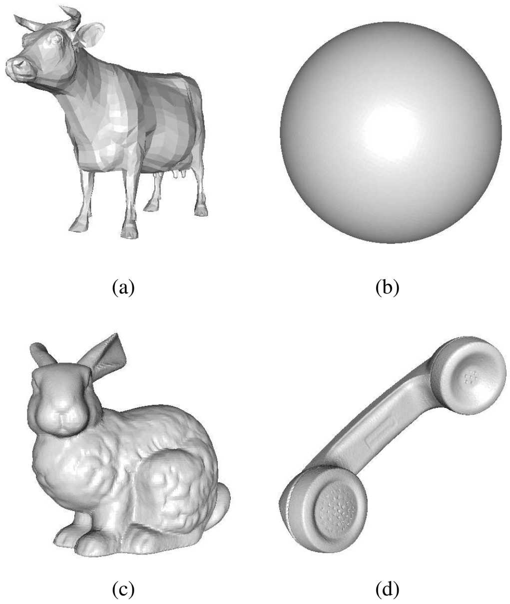

Figure 1: *a) cow; b) sphere; c) bunny; d) phone*

References

1. E. Puppo and R. Scopigno. Simplification, LOD and Multiresolution - Principles and Applications, *Eurographics'97 Tutorial Notes*, 16(3), 1997.
2. P. Heckbert and M. Garland. Multiresolution Modeling for Fast Rendering, *Proc. of Graphics Interface '94*, Banff Alberta Canada, 43–50, May, 1994.
3. H. Hoppe. Progressive Meshes, *Proc. of SIGGRAPH '96*, 99–108, 1996.
4. J. Ribelles, M. Chover, J. Huerta and R. Quirós. Multiresolution Ordered Meshes, *Proc. of 1998 IEEE Conference on Information Visualization*, London, 198–204, July, 1998.
5. H. Hoppe. Efficient Implementation of Progressive Meshes, *Computer & Graphics*, 22(1):27–36, 1998.
6. M. Garland and P. Heckbert. Surface Simplification Using Quadric Error Metric, *Proc. of SIGGRAPH '97*, 209–216, August, 1997.