Clustering Techniques for Out-of-Core Multi-resolution Modeling

Emanuele Danovaro[1,2], Leila De Floriani[1,2], Enrico Puppo[1], Hanan Samet[2]
[1] Department of Computer and Information Sciences (DISI) – University of Genova
[2] Department of Computer Science – University of Maryland at College Park

1 Introduction

Thanks to improvements in simulation tools, high resolution scanning facilities and multidimensional medical imaging, huge datasets are commonly available. Multi-resolution models manage the complexity of such data sets, by varying resolution and focusing detail in specific areas of interests. Since many currently available data sets cannot fit in main memory, the need arises to design data structures, construction and query algorithms for multi-resolution models which work in secondary memory.

Several techniques have been proposed in the literature for out-of-core simplification of triangle meshes, while much fewer techniques support multi-resolution modeling. Some such techniques only deal with terrain data [2, 8, 10, 11]. Techniques proposed in [3, 6, 7, 9, 14] have been developed for free-form surface modeling and most of them are based on space partitioning.

Our goal is to design and develop a general technique for irregularly distribuited data describing two and three-dimension scalar fields and free-form surfaces. In the spirit of our previous work, we define a general out-of-core strategy for a model that is independent of both the dimension and the specific simplification strategy used to generate it, i.e., the *Multi-Tessellation (MT)* [12, 5]. The MT consists of a coarse mesh plus a collection of refinement modifications organized according to a dependency relation, which guides extracting topologically consistent meshes at variable resolution. We have shown that the other multi-resolution data structures developed in the literature are specific instances of an MT. Thus, data structures optimized on the basis of a specific simplification operator, like edge collapse or vertex removal, could be derived from a general out-of-core MT.

The basic queries on a multi-resolution model are instances of *selective refinement*, which consists of extracting adaptive meshes of minimal size according to application-dependent requirements. We have first analyzed the I/O operations performed by selective refinement algorithms and designed and implemented a simulation environment which allows us to evaluate a large number of data structures for encoding a MT out-of-core. We have designed and developed more than sixty clustering techniques for the modifications forming a MT, which take into account their mutual dependency relations and their arrangement in space.

Based on the data structure selected through this investigation, we are currently developing an out-of-core prototype system for multi-resolution modeling which is independent of the way single modifications are encoded.

2 Clustering an MT

A Multi-Tessellation is made of a base mesh, a set of modifications and a dependency relation. A *modification* $u = (u^-, u^+)$ replaces a sub-mesh u^- with another, more refined, sub-mesh u^+ to locally increase resolution. Given a set of modifications acting on a base mesh Σ, we say that a modification u_i *directly depends* on another modification u_j if and only if u_i removes some of the cells introduced by u_j. The direct dependency relation defines a partial order on the set of modifications forming an MT.

Usually, the modifications consist of a small number of cells (triangles/tetrahedra), and thus we need to cluster them to fill up a disk page. The major issue here is that the dependency relation is a partial order, and thus it cannot be described by a tree, but rather by a Directed Acyclic Graph (DAG). Moreover, selective refinement algorithms are not based on classical graph traversal strategies, therefore classical out-of-core structuring and traversal techniques cannot be applied here.

Selective refinement queries can be classified into queries at a *uniform* Level-Of-Detail (LOD) and queries at a *variable* LOD. Uniform LOD queries require a uniform resolution on the whole shape, while variable LOD ones require resolution focused on a region of interest. View-dependent queries are also examples of variable-resolution queries. An analysis of selective refinement queries and of the shape of the DAG describing the MT has suggested us two classes of clustering strategies: one based on different sorting of the modifications in the MT (some derived by DAG traversals); and the other based on space-based grouping.

Uniform-resolution queries have suggested partitioning the MT into layers, where each successive layer should guarantee a uniform increase of resolution. We have defined and implemented the following techniques for grouping the modifications in an MT according to sorting criteria:
- Approximation error (`Err`);
- Layer: shortest path from the root (`Lyr`); Level: longest path from root (`Lev`); Distance: average length of paths from the root (`Ly2`);
- Depth-first (`DFS`) and Breadth-first (`BFS`);
- Multi-resolution visit - depth first (`GrD`) and Multi-resolution visit - breadth first (`GrB`): similar to depth-first or breadth-first DAG traversal, respectively, but before adding a modification u, all ancestors of u are visited recursively if they have not been visited before. These criteria simulate the strategies used in a selective refinement algorithm.

Variable-resolution queries have suggested grouping according to a space clustering technique. We have defined and implemented to spatial grouping strategies. The first strategy uses an extent-based aggregation technique, the *R*-tree*, which associate bounding boxes to spatial objects and recursively aggregate k of them into a minimal bounding box [1]. This is a dimension-independent technique, which works with d-dimensional boxes. We have applied it by associating a bounding box to each modification in the MT or to a modification extruded along a further dimension, representing, for instance the approximation error, the layer or the distance.

The second strategy is based on a *Point Region k-d (PR k-d* tree for partitioning in space combined with a PK-tree [13] as grouping mechanism. A PR k-d tree is a dimension-independent spatial index based on the recursive subdivision of the domain containing a set of points into two halves. The subdivision process cycles through different dimensions in a predefined and constant order. At each step a block is subdivided in its mid-point. We associate with each modification its centroid, and we construct the PR k-d tree according to the coordinates of centroids. We have performed several tests in higher dimensions by adding the approximation error or the layer or the distance from the root as an additional dimension. A PK-tree is

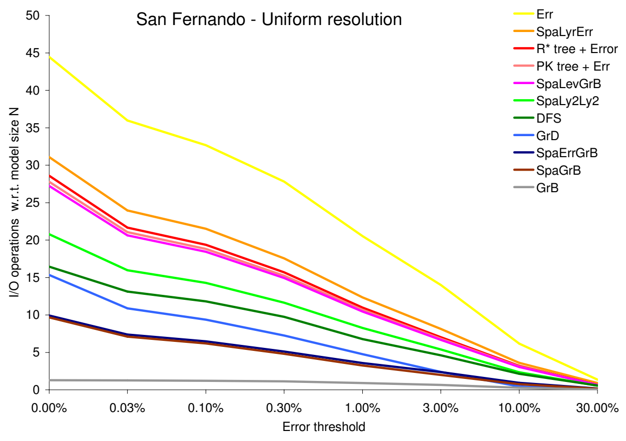

Figure 1: Ratio between the number of I/O operations performing queries at uniform resolution and the number of disk blocks required to store the model.

constructed by applying a bottom-up grouping process to the nodes of a tree T. Nodes belonging to T are grouped into clusters until the minimum occupancy k has been reached. During the grouping process empty leaves of the tree T are removed. An interesting property of the PR k-d tree grouped according to a PK-tree, that we called a *PK PR k-d tree*, is that each node has a minimum of k children and a maximum of $2\cdot(k-1)$, regardless of the dimensionality of the space. This guarantees that disk blocks are at least half-full.

Finally, we have designed and implemented a class of strategies which combine space subdivision with DAG subdivision (according to depth). At a lower resolution, we are interested to have clusters that span the whole domain, while, as the resolution increases, we are looking for clusters associated with finer space subdivisions. In order to achieve this goal, we have developed techniques that interleave the effect of a sorting rule and the effect of a space partitioning rule similar to a PR k-d tree. A description of such techniques can be found in [4].

3 Experimental Results

We have performed our experiments on both terrain and volume data sets. Here, we present only results on a tetrahedral data set, representing the effect of an earthquake in the San Fernando valley (CA), which consists of 2,067,739 tetrahedra. A complete set of experiments can be found in [4]. During our simulations, we have artificially reduced the amount of available core memory. This forces the out-of-core prototype to load only a subset of disk blocks at a time.

Figure 1 compares some of the clustering techniques based on sorting, space grouping and some combination of sorting and space partitioning, namely:
- space partitioning on three coordinates, and `GrB` sorting (`SpaGrB`);
- space partitioning on approximation error and three coordinates, and `GrB` sorting (`SpaErrGrB`);
- space partitioning on layer and three coordinates, and `Err` sorting (`SpaLyrErr`);
- space partitioning on distance and three coordinates, and `Ly2` sorting (`SpaLy2Ly2`);
- space partitioning on level and three coordinates, and `GrB` sorting (`SpaLevGrB`);
- space partitioning on approximation error and three coordinates, and grouping (`PK PR kd tree + Error`);
- space grouping with an R*-tree built according to a bounding box based on 3D spatial extent and approximation error (`R* tree + Error`).

It can be easily seen that `GrB` outperforms the other clustering techniques, and this is true for all experiments we have performed on different data sets and with the sixty clustering strategies [4]. This is no surprise for uniform-LOD queries, while it is rather unexpected for queries at variable resolution. It is also interesting to note that, even with a small cache (about 1% of the size of the whole model), a clustering technique based on `GrB` exhibits a very limited overhead, compared to loading the whole model.

Acknowledgments

This work has been supported by MIUR projects SPADA-WEB and Lima3D, by the European Network of Excellence AIM@SHAPE under contract number 506766, and by the National Science Foundation under Grant EIA-00-91474 as well as Microsoft Research.

References

[1] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: an efficient and robust access method for points and rectangles. In *Proceedings ACM SIGMOD Conference*, pages 322–331, Atlantic City, NJ, June 1990. ACM Press.

[2] P. Cignoni, F. Ganovelli, E. Gobbetti, F. Marton, F. Ponchio, and R. Scopigno. Planet–sized batched dynamic adaptive meshes (P-BDAM). In *Proceedings IEEE Visualization*, pages 147–155, Seattle, WA, USA, October 2003. IEEE Computer Society Press.

[3] P. Cignoni, F. Ganovelli, E. Gobbetti, F. Marton, F. Ponchio, and R. Scopigno. Adaptive TetraPuzzles – efficient out-of-core construction and visualization of gigantic polygonal models. *ACM Transactions on Graphics*, 23(3), August 2004.

[4] E. Danovaro, L. De Floriani, E. Puppo, and H. Samet. Multi-resolution out-of-core modeling of terrain and geological data. In *Symposium on Advances in Geographic Information Systems*, New York, NY, USA, November, 4-5 2005. ACM Press. Accepted.

[5] L. De Floriani, P. Magillo, and E. Puppo. Building and traversing a surface at variable resolution. In *Proceedings IEEE Visualization 97*, pages 103–110. IEEE Computer Society, October 1997.

[6] C. DeCoro and R. Pajarola. XFastMesh: Fast view-dependent meshing from external memory. In *Proceedings IEEE Visualization 2002*, pages 263–270, Boston, MA, October 2002. IEEE Computer Society.

[7] J. El-Sana and Y.-J. Chiang. External memory view-dependent simplification. *Computer Graphics Forum*, 19(3):139–150, August 2000.

[8] H. Hoppe. Efficient implementation of progressive meshes. *Computer & Graphics*, 22(1):27–36, 1998.

[9] P. Lindstrom. Out-of-core construction and visualization of multiresolution surfaces. In *ACM SIGGRAPH 2003 Symposium on Interactive 3D Graphics*, pages 93–102, Monterey, California, April 2003. ACM Press.

[10] P. Lindstrom and V. Pascucci. Terrain simplification simplified: a general framework for view-dependent out-of-core visualization. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):239–254, 2002.

[11] P. Magillo. *The MT (Multi-Tessellation) Package*. Department of Computer Sciences (DISI), University of Genova, Italy, January 2000.

[12] E. Puppo. Variable resolution terrain surfaces. In *Proceedings Eight Canadian Conference on Computational Geometry, Ottawa, Canada*, pages 202–210, August 12-15 1996.

[13] W. Wang, J. Yang, and R. Muntz. PK-tree: a spatial index structure for high dimensional point data. In K. Tanaka and S. Ghandeharizadeh, editors, *Proceedings 5th International Conference on Foundations of Data Organization and Algorithms (FODO)*, pages 27–36, Kobe, Japan, November 1998.

[14] S.E. Yoon, B. Salomon, R. Gayle, and D. Manocha. Quick-VDR: Out-of-core view-dependent rendering of gigantic models. *IEEE Transactions on Visualization and Computer Graphics*, 11(4):369–382, 2005.