An Optimized, Grid Independent, Narrow Band Data Structure for High Resolution Level Sets

Michael Bang Nielsen[*]
University of Aarhus

Ken Museth[†]
GG[‡]@Linköpings University

1 Introduction

Level sets have recently proven successful in many areas of computer graphics including water simulations[Enright et al. 2002] and geometric modeling[Museth et al. 2002]. However, current implementations of these level set methods are limited by factors such as computational efficiency, storage requirements and the restriction to a domain enforced by the convex boundaries of an underlying cartesian computational grid. Here we present a novel very memory efficient narrow band data structure, dubbed the Sparse Grid, that enables the representation of grid independent high resolution level sets. The key features our new data structure are

- Both memory usage and computational efficiency scales linearly with the size of the interface.
- The values in the narrow band can be compressed using quantization without compromising visual quality.
- The level set propagation is independent of the boundaries of an underlying grid. Unlike previous methods that use fixed computational grids with convex boundaries our Sparse Grid can expand and/or contract dynamically in any direction with non-convex boundaries.
- Our data structure generalizes to any number of dimensions.
- Our flexible data structure can transparently be integrated with the existing finite difference schemes typically used to numerically solve the level set equation on fixed uniform grids.

2 Data Structure

Previously proposed methods for localizing level set computations[Peng et al. 1999]require the entire 3D grid and additional data structures representing the narrow band to be present in memory. However, our Sparse Grid data structure implements localized level set computations on top of a dynamic narrow band storage scheme to obtain both a time and space-efficient data structure.

Figure 1 shows a human head and a slice of its corresponding 3D Sparse Grid representation. The values of all grid points in the narrow band, shown in red and green, are explicitly stored and may additionally be quantized to further reduce the memory footprint. The (x,y,z) index-vector of each grid point in the narrow band must also be stored, however explicit storage does not scale well. Instead we exploit the connectivity of the narrow band to develop an efficient index-vector storage scheme with a compression. This compact storage scheme is defined recursively in the dimensions of the grid and allows the Sparse Grid to easily generalize to any dimension. For a 3D Sparse Grid, the scheme stores: 1) The start and end z-index of each connected component in the z-direction. 2) The start and end y-index of each connected component in the y-direction contained in the projection of the 3D narrow band onto the x-y plane. 3) The start and end x-index of each connected component in the x-direction contained in the recursive projection of the 3D narrow band onto the x-axis. The actual grid point index-vector of the values in the narrow band are decoded on the fly. Sequential access to all grid points in the Sparse Grid can be done in linear time, thus providing constant time access to each grid point on average. Level set propagations typically require information about the gradient, curvature etc. in each grid point, which requires knowledge of neighboring points. Constant time access to neighboring grid points within a stencil is provided by iterating a stencil of iterators through the narrow band. Random access is logarithmic.

The N-dimensional Sparse Grid enables the narrow band to be rebuilt in linear time using a fast algorithm that takes advantage of the information about connected components readily available in the storage format.

[*]e-mail: bang@daimi.au.dk
[†]e-mail: museth@acm.org
[‡]http://www.gg.itn.liu.se

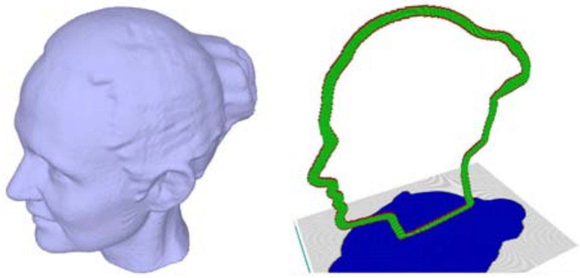

Figure 1: Human head and slice of Sparse Grid representation.

3 Evaluation

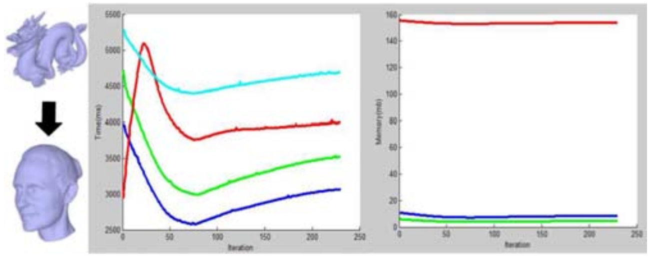

Figure 2: Morph time and memory usage on a $193 \times 356 \times 251$ grid.

Figure 2 shows the time (left) and memory (right) usage for a level set morph computed with the method of Peng(cyan), the method of Peng improved with a memory efficient localized narrow band update (red), the Sparse Grid (blue) and the Sparse Grid with a 16 bit quantization (green). The Sparse Grid methods are mostly faster and use far less memory. The methods of Peng use the same amount of memory. The improved method of Peng (red) does not ensure cache coherency which explains the rapid increase in running time at the beginning. We plan to apply the Sparse Grid to several research areas in computer graphics in the future.

References

ENRIGHT, D., MARSCHNER, S., AND FEDKIW, R. 2002. Animation and rendering of complex water surfaces. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM Press, 736–744.

MUSETH, K., BREEN, D. E., WHITAKER, R. T., AND BARR, A. H. 2002. Level set surface editing operators. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM Press, 330–338.

PENG, D., MERRIMAN, B., OSHER, S., ZHAO, H., AND KANG, M. 1999. A pde-based fast local level set method. *J. Comput. Phys. 155*, 2, 410–438.