Time series AMR data representation for out-of-core interactive visualization

W. Alexandre-Barff†[1], H. Deleau[1], J. Sarton[3], F. Ledoux[2] and L. Lucas[1]

[1] Université de Reims Champagne-Ardenne, LICIIS, LRC DIGIT, France
[2] CEA, DAM, DIF, LRC DIGIT, F-91297 Arpajon, France
[3] Université de Strasbourg, ICube, UMR-CNRS 7357, France

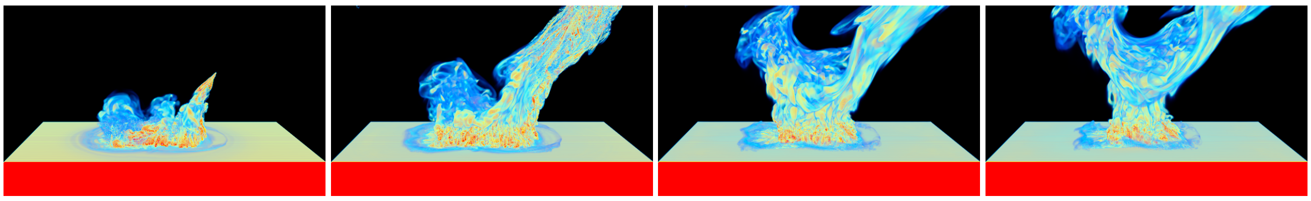
Figure 1: *Visualization of four time steps at Λ^0_{63}, Λ^0_{145}, Λ^0_{229} and Λ^0_{269} on Deep Ocean Water AMR time series*

Abstract

Time-varying Adaptive Mesh Refinement (AMR) data have become an essential representation for 3D numerical simulations in many scientific fields. This observation is even more relevant considering that the data volumetry has increased significantly, reaching petabytes, hence largely exceeding the memory capacities of the most recent graphics hardware. Therefore, the question is how to access these massive data - AMR time series in particular - for interactive visualization purposes, without cracks, artifacts or latency. In this paper, we present a time-varying AMR data representation to enable a possible fully GPU-based out-of-core approach. We propose to convert the input data initially expressed as regular voxel grids into a set of AMR bricks uniquely identified by a 3D Hilbert's curve and store them in mass storage.

CCS Concepts

• ***Human-centered computing*** $\rightarrow$ ***Scientific visualization;***

1. Introduction

Numerical simulation is defined as a mathematical process to study through dedicated models real-world complex temporal phenomena. It allows scientists to enhance understanding and gain critical insight for the interpretation and diagnosis of the latter. Among all the approaches currently in use, the ones based on AMR are the most suitable for sparse computational domains study. The AMR principle seeks to combine the simplicity of structured grids with the advantages of local refinement to obtain a multi-resolution hierarchy of cells. These techniques can be divided into two categories: *block-structured* (BS-AMR) and *tree-based* (TB-AMR). However, interactive navigation inside such time series AMR involves designing scalability methods to handle large-scale data. Our approach aims to achieve high-quality visualization on a single workstation by relying on an out-of-core approach which addresses entire massive datasets from high-performance computing devices such as current GPUs. To fulfill this it is necessary to provide a suitable data representation to address independently any time-series AMR blocks. We base our work on BS-AMR multiresolution bricking and 3D Hilbert's curve indexing methods for addressing purposes.

2. Related Works

Out-of-core data management defines the class of techniques used to handle data that are too large to fit entirely into the main memory of the unit in charge of their processing. These methods were made to answer the unbalanced memory capacity between mass storage and the system. These same algorithms can be used in scientific visualization to handle the insufficiency of GPU memory as opposed to the increase of computational power. Among all the established out-of-core approaches, Sarton et al. proposition [SCRL20] introduce a data structure based on a caching strategy with a virtual memory addressing system coupled to efficient parallel management on GPU to provide efficient access to data in interactive time. Our integrating method proposes to extend this approach in

† This research work is supported by the LRC DIGIT, a joint lab between CEA-DAM Ile de France and URCA. Authors would like also to thank B. Lévy from INRIA and J. Patchett from LANL for providing datasets

DOI: 10.2312/egp.20221014

a general-purpose framework allowing us to visualize and process interactively time-dependent AMR datasets on the GPU.

3. Our Proposition

Performed once per dataset, this stage first converts each sequence of regular 3D grids into a multi-resolution pyramid of AMR independent blocks called "bricks". Thus, the computational domain is covered by a set of coarse, structured subgrids $\Gamma^0_{l=0...n} \subset R^3$. The union of these subgrids at each time $t \in N$ is called the root level Λ^0_t and defined by $\cup^n_{m=0}\Gamma^0_m$. In the same way, the union of all l subgrids $\Gamma^l_{m=0,1,2...}$ is called refinement level Λ^l_t or just level l at timestamp t and by construction these levels for the same timestamp satisfies the relationship $\Lambda^{l+1}_t \subseteq \Lambda^l_t \subseteq \Lambda^0_t$. This conversion scheme is largely derived from the work of [MAS15]. Results produced by this method return a set of AMR bricks always containing the same number of voxels with different spatial resolution but the same memory footprint.

SFC indexing: Once all the AMR brick data is obtained, we use a *Space-Filling Curve* (SFC) path to address them uniquely and individually. Defined in a unit cube, this 3D SFC allows us to convert - as a mapping function - 3D position into a single Id. We chose Hilbert's curve here because of its property to preserve the locality distance between near points. The last part of this preprocessing stage is responsible to save and organize the produced data. We store each AMR brick data scalars into individual binary files in addition to a supplementary *JavaScript Object Notation* (JSON) file at the top level of the hierarchy.

BAT: The main component of our integrating method called *Brick Addressing Table* (BAT) is an AMR bricks array from a single time step. It is an ascending order array of every AMR brick data from the coarsest to the finest level. In addition, they are also organized between all bricks of the same refinement level by their own respective SFC index in ascending order. This ordering is the key feature of the BAT, as it enables a binary search of an AMR brick by their SFC index. Once we create a BAT for every time step of the AMR time series data, we use an upper-level of our out-of-core ready approach to handle the collection of BAT alongside the data cache management.

TS-BAT: Our out-of-core ready approach called *Time Step Brick Addressing Table* (TS-BAT) is an array of time-ordered BAT and the only information stored on GPU with the data cache at all times. It can be seen as an upper-level of the data cache with every entry addressing a single AMR brick in the whole time series data and guaranteeing the out-of-core mechanism is threefold: ***i)*** **Virtualization mechanism** by creating successive upper-level arrays of *Virtual Bricks* (VB) one over the other on top of the data cache. Each VB successively addresses a subdivided lower level VB, until reaching the data cache and stores the position of the addressed VB with a flag to know if the AMR brick to which It belongs exists in the data cache. ***ii)*** **GPU interface encoding** by translating the voxel request coming from the end-user application to navigate through the TS-BAT with a binary search on each successive VB level. ***iii)*** **Data cache management** by utilizing an array of entries called *Least Recently Used* system (LRU) on each array level of VB to ensure the update of every needed AMR brick from the data cache. Every LRU has the same number of entries as VB from the array level it handles. Each time an AMR brick belonging to an array level is requested, the corresponding entry in the LRU is orderly moved up. The most recently requested AMR brick is at the top-order to the bottom for the last requested one. Once a requested voxel is not in the data cache, the TS-BAT sends a request to the CPU. To swap the newest brick of data, the TS-BAT discards from the data cache and every corresponding array level the addressing AMR brick. It picks the bottom-order brick of each corresponding LRU to discard and replace it with the newest AMR brick in each corresponding LRU.

4. Preliminary Results and Discussion

To illustrate our results, we used two datasets defined as a sequence of i) dataset 1: 269 volumes of 460 $\times$ 280 $\times$ 240 voxels encoded on 128bits (4 scalar fields - material Id, temperature, volume fraction of water and volume fraction of asteroid). It represents the study of asteroid impacts in deep ocean water. ii) dataset 2: 311 volumes of 1000 $\times$ 1000 $\times$ 1000 binary voxels representing a free-surface fluid simulated by a semi-discrete partial optimal transport algorithm. Each of these series has been preprocessed by setting the subdivision level to 3. To validate our approach initially on CPU, we have integrated our code in the OpenVKL [Int20] AMR volume rendering engine. From these early results, we have achieved an interactive volume visualization of our AMR data representation, which reinforces our integrating method for an out-of-core ready approach. Lastly, we reached an average preprocessing time of around 9min for dataset 1 with a memory size of around 133Go and over a 2,5hours for dataset 2 with a 1,2To memory size. In addition, our preprocessing shows potential for in-situ visualization as it averages a 30s time for a single step from our biggest dataset.

5. Conclusion

In this paper, we proposed a 3D time-dependent AMR data representation for out-of-core ready approach interactive volume visualization. Our global pipeline starts from the AMR data representation during a preprocessing step that converts regular voxels grids time series into AMR brick per time step uniquely identified by a 3D Hilbert's curve. To the integrating method to an out-of-core ready approach to address the entirety of the time series which exceeds GPU VRAM capacity by uploading on the fly the only AMR brick data needed. Our approach differs from [SCRL20] by preprocessing AMR time series data and 3D Hilbert's curve method instead of natural indexing. Our first results on CPU are promising and demonstrate the viability of our approach. However, at this stage, it may be noted the need to have a GPU SFC encoding method. As well as confirms that our TS-BAT fully operates on an entire dataset before implementing it on GPU.

References

[Int20] INTEL: Openvkl high performance volume kernels. https://www.openvkl.org/, 2020. Accessed: 2022-02-15. 2

[MAS15] M. ADAMS P. COLELLA D. G. J. J. N. K. T. L. D. M. P. M. D. M. P. S. T. S., STRAALEN B. V.: *Chombo Software Package for AMR Applications - Design Document*. Report LBNL-6616E, Lawrence Berkeley National Laboratory Technical Report, 2015. 2

[SCRL20] SARTON J., COURILLEAU N., REMION Y., LUCAS L.: Interactive Visualization and On-Demand Processing of Large Volume Data: A Fully GPU-Based Out-of-Core Approach. *IEEE Transactions on Visualization and Computer Graphics 26*, 10 (Oct. 2020), 3008–3021. doi:10.1109/TVCG.2019.2912752. 1, 2