Journal of Computational Physics 345 (2017) 91–110

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Simulation of tsunamis generated by landslides using adaptive mesh refinement on GPU

M. de la Asunción *, M.J. Castro

Dpto. Análisis Matemático, Estadística e Investigación Operativa, y Matemática Aplicada, Universidad de Málaga, Spain

ARTICLE INFO

Article history:
Received 16 November 2016
Received in revised form 4 May 2017
Accepted 9 May 2017
Available online 12 May 2017

Keywords:
Tsunamis generated by landslides
Finite volume methods
Shallow water equations
Adaptive mesh refinement
CUDA

ABSTRACT

Adaptive mesh refinement (AMR) is a widely used technique to accelerate computationally intensive simulations, which consists of dynamically increasing the spatial resolution of the areas of interest of the domain as the simulation advances. During the last years there have appeared many publications that tackle the implementation of AMR-based applications in GPUs in order to take advantage of their massively parallel architecture. In this paper we present the first AMR-based application implemented on GPU for the simulation of tsunamis generated by landslides by using a two-layer shallow water system. We also propose a new strategy for the interpolation and projection of the values of the fine cells in the AMR algorithm based on the fluctuations of the state values instead of the usual approach of considering the current state values. Numerical experiments on artificial and realistic problems show the validity and efficiency of the solver.

1. Introduction

A tsunami is one of the most dangerous natural phenomena. It is usually generated by a big earthquake or landslide, and it may have a big impact on infrastructures and human lives at coastal areas. For this reason it is very important to have very fast simulators in order to analyze the impact that a tsunami may have in certain zones, so that appropriate prevention measures could be taken.

The shallow water equations [1] are typically used to simulate the evolution of a tsunami, but a great computational power is needed to perform these simulations in a reasonable time due to the big dimensions of the spatial and temporal domains. One way to speed up these simulations is by implementing the solver using a GPU programming language, such as CUDA [2] or OpenCL [3], thus taking advantage of the massively parallel architecture of the modern graphics cards.

Another way to improve the efficiency of the simulations is using adaptive mesh refinement (AMR). This technique consists of dynamically increasing the spatial resolution of the regions of interest of the domain as the simulation progresses, while the rest of the domain is kept at low resolution, thus obtaining better runtimes and similar results compared to increasing the spatial resolution of the entire domain. In the case of a tsunami, one may have high spatial resolutions at the main propagating waves of the tsunami and at the coastal areas when the tsunami hits them, while low resolutions can be applied to the rest of the domain.

* Corresponding author. Fax: +34 952 131894.
E-mail address: marcah@uma.es (M. de la Asunción).

http://dx.doi.org/10.1016/j.jcp.2017.05.016
0021-9991/

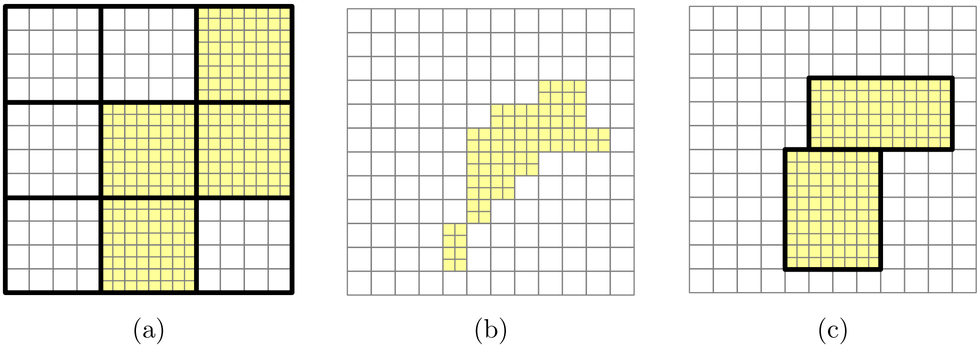

Fig. 1. Kinds of AMR: (a) Block-based, (b) Cell-based, (c) Patch-based.

We can differentiate three kinds of AMR, depicted in Fig. 1:

- *Block-based*: In this approach the spatial mesh is divided into fixed blocks, and one block is refined if a certain condition is achieved, for example if any cell inside the block is refined. The advantages of the block-based AMR are the processing using structured meshes and relatively simple data structures, but it may lead to an over-refined mesh.
- *Cell-based*: In this case only the cells that need to be refined are refined. All the cells are typically processed individually and the advantage of processing structured meshes is lost.
- *Patch-based*: In this strategy the refined cells are dynamically grouped together to form rectangular patches. This approach tries to obtain the benefits of the block-based and the cell-based AMR, that is, having structured meshes without over-refining the mesh. However, a clustering algorithm is needed to create the different patches.

AMR was first proposed by M.J. Berger and collaborators in a series of well-known papers [4,5]. They introduced a patch-structured AMR approach developed within the finite volume framework and later used extensively for astrophysical applications. Other interesting developments have been presented in [6], where the first high order AMR algorithms based on WENO finite volume schemes were introduced. Applications of AMR techniques in the field of shallow water equations have been reported, for instance, in [7] and [8]. GeoClaw [8,9] is perhaps the most well known code on geophysical flows, including tsunamis, dam breaks and storm surges, that incorporates AMR. Other interesting implementations of adaptive mesh refinement are based on the so-called quadtree/octree refinement; for an overview of these techniques in different contexts see [10,11]. AMR techniques have also been successfully implemented with Central WENO (CWENO) schemes, for example in [12,13]. The first implementation of a high order ADER-WENO finite volume scheme with AMR was proposed in [14] and [15] for conservative and non-conservative hyperbolic PDEs, respectively, in two and three space dimensions. This implementation followed a cell-based approach and it was extended to special relativistic hydrodynamics (RHD) and magnetohydrodynamics (RMHD) in [16].

In the framework of Discontinuous Galerkin (DG) schemes there is a significant number of papers using AMR or *hp*-adaptivity. Two well-known early series of papers on *hp*-adaptive DG schemes are due to Baumann and Oden [17,18], and Houston, Schwab and Süli [19–21]. Other relevant results have been obtained in [22]. There also exist several CPU packages to help developers to incorporate adaptive mesh refinement techniques into their applications, such as PARAMESH [23], BoxLib [24], Chombo [25] or SAMRAI [26].

The number of AMR-based applications accelerated using GPUs has greatly increased during the last years. Wang et al. [27] port to GPU the hydrodynamics code Enzo [28] using patch-based AMR, where the hydrodynamics solver is implemented in GPU and the creation of the grid hierarchy is performed on the CPU. GAMER [29] is a hybrid CPU-GPU astrophysics code that uses block-based AMR with blocks of size 8×8 cells. Burstedde et al. [30] present a hybrid CPU-GPU code for seismic wave propagation using block-based AMR, where the wave propagation solver runs on the GPU and the AMR operations execute on the CPU. Arce and Aoki [31] present a multi-GPU implementation to simulate tsunamis by means of shallow water equations using patch-based AMR and space-filling curves. Beckingsale et al. [32] extend the SAMRAI library allowing AMR simulations to be executed on GPU, and use this extension to implement a patch-based AMR hydrodynamics code in GPU, where the operations to manage the grid hierarchy are run on the CPU. Sætra et al. [33] describe a GPU implementation to solve the shallow water equations using the second-order Kurganov–Petrova scheme [34] employing a block-based AMR approach, where the obtaining of the coordinates of the new grids is the only part of the algorithm performed on the CPU. Lastly, Wahib and Maruayama [35] propose a block-based AMR implementation on GPU where only the managing of the AMR metadata is carried out on the CPU, and they apply it to a hydrodynamics and a shallow water solver.

In this work we propose a CUDA implementation of a patch-based AMR algorithm to simulate tsunamis generated by landslides using the two-layer Savage–Hutter type model described in [36], where only the calculation of the new sub-meshes is performed on the CPU. To the best of our knowledge, this is the first AMR-based application implemented on GPU using a two-layer shallow water system for the simulation of tsunamis generated by landslides. Furthermore, in the