Parallelization of the Finite-Difference Time-Domain Method for acoustic room simulation

Jose J. Lopez, Diego Carnicero, Nestor Ferrando, Joaquín Cerdá

jjlopez@dcom.upv.es, diecarvi@teleco.upv.es, nesferjo@upvnet.upv.es, joacerbo@eln.upv.es

ITEAM, Universidad Politécnica de Valencia

In the last years, the Finite-Difference Time-Domain (FDTD) approximation method has been introduced into acoustics to solve field problems numerically. However, the huge computer power needed in large scale problems when full band simulations are desired has delayed its applications in the different fields of acoustics. For example, in room simulation, FDTD provides an accurate sound field rendering, predicting the nature of the wave propagation in complex shaped rooms. However, other less accurate geometric methods, as ray tracing or the image method, are usually employed because their implementation demands less computing resources.

The FDTD method, due to its recursive nature is easily parallelizable. Different experiences using multi-core and multi-CPU architectures in PC have been already carried out by the authors and by others. However, this parallelization can only be extended typically to 8 threads (2 CPU with 4 cores each one). Despite the obtained results are successful, a massive parallelization is interesting to solve even bigger problems (i.e. large rooms up to high freq.).

Recently, the power of Graphics Processing Units (GPU) has been increased and the introduction of open development libraries has made their employment very easy, opening new possibilities for implementing massively parallel algorithms.

In this paper, we present an efficient implementation of a FDTD algorithm over GPUs applied to room simulation. The FDTD algorithm has been implemented using the library CUDA from NVidia, paying special attention in its efficiency by taking into account all the singularities of the GPU architecture.

The resulting algorithm has been tested in tested in different room simulation experiments, showing a high degree of precision and a notable acceleration compared with multiple-core CPU. Specifically, the speed up obtained in this work outperforms previous published papers, obtaining speeds up around 15x compared with state of the art quad-core CPUs.