2Pb2-1

Design and Implementation of High-order FDTD Method for Room Acoustics

Tan Yiyu[†], Toshiyuki Imamura, Masaaki Kondo
RIKEN Center for Computational Science
7-1-26 Minatojima-minami-machi, Chuo-ku, Kobe, Hyogo, Japan

1. Introduction

FDTD method has already become an essential method in room acoustics because of its high accuracy, easy implementation and parallelism. It is computation-intensive and memory-intensive as the problem size is increased because of the oversampling in spatial grids to suppress the inherent numerical dispersion. To date, many works have already done at algorithmic level to reduce the inherent dispersion and oversampling in FDTD method [1-4]. However, these methods still require high computation capability and memory bandwidth.

On the other hand, GPUs and FPGAs were applied to speed up computation in sound field rendering in recent years because of their much higher parallelism over traditional general-purpose processors [5-7]. In particular, latest FPGAs contain thousands of hardened floating-point arithmetic units, several Megabytes of on-chip block memories, and millions of reconfigurable logic blocks. All these on-chip hardware resources may be applied to directly implement sound wave equation to accelerate computation in contrast with software-based solutions in GPUs and general-purpose processors. Therefore, FPGAs has become promising in sound field rendering. In this research, an FPGA-based accelerator is developed to speed computation in sound field rendering with the high-order FDTD scheme.

2. High-order FDTD Scheme

In the high-order FDTD scheme, high-order approximation based on the Lagrange polynomial fitting [8] can be employed to approximate the second-order partial derivative instead of the traditional second-order center difference method in spatial domain in sound wave equation. In order to reduce computation, the second-order center difference method is applied on the time domain. For example, in the 4th-order scheme, the second-order partial derivative is approximated by using equation (1) [2].

$$\begin{aligned}
\frac{\partial^2 P}{\partial t^2} &= \frac{P^{n-1}_{i,j,k} - 2P^{n}_{i,j,k} + P^{n+1}_{i,j,k}}{\Delta t^2} \\
\frac{\partial^2 P}{\partial x^2} &= \frac{-\frac{1}{12}(P^{n}_{i-2,j,k} + P^{n}_{i+2,j,k}) + \frac{4}{3}(P^{n}_{i-1,j,k} + P^{n}_{i+1,j,k}) - \frac{5}{2}P^{n}_{i,j,k}}{\Delta x^2} \qquad (1) \\
\frac{\partial^2 P}{\partial y^2} &= \frac{-\frac{1}{12}(P^{n}_{i,j-2,k} + P^{n}_{i,j+2,k}) + \frac{4}{3}(P^{n}_{i,j-1,k} + P^{n}_{i,j+1,k}) - \frac{5}{2}P^{n}_{i,j,k}}{\Delta y^2} \\
\frac{\partial^2 P}{\partial z^2} &= \frac{-\frac{1}{12}(P^{n}_{i,j,k-2} + P^{n}_{i,j,k+2}) + \frac{4}{3}(P^{n}_{i,j,k-1} + P^{n}_{i,j,k+1}) - \frac{5}{2}P^{n}_{i,j,k}}{\Delta z^2}
\end{aligned}$$

Letting $\Delta x = \Delta y = \Delta z = \Delta l$ and inserting equation (1) into the sound wave equation as shown in equation (2), equation (3) is derived to update sound pressures for the fourth-order scheme [9].

$$\frac{\partial^2 P}{\partial t^2} = c^2\left(\frac{\partial^2 P}{\partial x^2} + \frac{\partial^2 P}{\partial y^2} + \frac{\partial^2 P}{\partial z^2}\right) \qquad (2)$$

$$\begin{aligned}
P^{n+1}_{i,j,k} = \chi^2[&-\frac{1}{12}(P^{n}_{i-2,j,k} + P^{n}_{i+2,j,k} + P^{n}_{i,j-2,k} + P^{n}_{i,j+2,k} \\
&+ P^{n}_{i,j,k-2} + P^{n}_{i,j,k+2}) + \frac{4}{3}(P^{n}_{i-1,j,k} + P^{n}_{i+1,j,k} + P^{n}_{i,j-1,k} \qquad (3) \\
&+ P^{n}_{i,j+1,k} + P^{n}_{i,j,k-1} + P^{n}_{i,j,k+1})] + (2 - \frac{15}{2}\chi^2)P^{n}_{i,j,k} - P^{n-1}_{i,j,k}
\end{aligned}$$

where $\chi = c\Delta t/\Delta l$ is the Courant number. The updated equation of other higher order scheme can be obtained by using the similar derivation procedure, such as equation (4), which is the updated equation of the 6th-order scheme [9].

$$\begin{aligned}
P^{n+1}_{i,j,k} = \chi^2[&\frac{1}{90}(P^{n}_{i-3,j,k} + P^{n}_{i+3,j,k} + P^{n}_{i,j-3,k} + P^{n}_{i,j+3,k} \\
&+ P^{n}_{i,j,k-3} + P^{n}_{i,j,k+3}) - \frac{3}{20}(P^{n}_{i-2,j,k} + P^{n}_{i+2,j,k} + P^{n}_{i,j-2,k} \qquad (4) \\
&+ P^{n}_{i,j+2,k} + P^{n}_{i,j,k-2} + P^{n}_{i,j,k+2}) + \frac{3}{2}(P^{n}_{i-1,j,k} + P^{n}_{i+1,j,k} + P^{n}_{i,j-1,k} \\
&+ P^{n}_{i,j+1,k} + P^{n}_{i,j,k-1} + P^{n}_{i,j,k+1})] + (2 - \frac{49}{6}\chi^2)P^{n}_{i,j,k} - P^{n-1}_{i,j,k}
\end{aligned}$$

Equations (3) and (4) indicate that a grid and its neighbor grids along axes are required to compute its sound pressure.

3. System Design

Sound field rendering with FDTD methods is

---
tan.yiyu@riken.jp

memory-intensive. To reduce the required memory bandwidth, a large sound space with $M \times N \times Nz$ grids is firstly decomposed into small spatial blocks and each block has $Cx \times Cy \times Nz$ grids. A spatial block is further partitioned into x-y planes along the z dimension. Computations are performed on the basis of planes inside a block while they are carried out along the x dimension inside a plane. When a spatial block is computed and results are stored in on-chip block RAMs inside an FPGA, computation for the same spatial block at next time steps is performed without waiting for computations of all grids are completed. Therefore, the external memory accesses are reduced and computation at several time steps is carried out in parallel.

The system diagram is illustrated in Fig. 1. The system consists of the Data input module, computation engine, and Data output module. The Data input module streams data plane by plane of a spatial block from the external DDR memory on the FPGA card, and feeds data to the computation engine. The computation engine consists of several processing elements (PEs). Each PE computes sound pressures of grids in a spatial block at a time step, and several PEs are cascaded to compute sound pressures of grids in the same spatial block at continuous time steps. Therefore, accesses to external memory are reduced and computation is accelerated because sound pressures of grids in a spatial block at several time-steps are computed concurrently. As shown in Fig. 2, A PE computes sound pressure of a grid according to its position, incidence, and values of its neighbours at previous time steps. The computed results are sent to the neighbour PE except the final PE, in which they are written to the external memory through the Data output module.

4. Performance Estimation

The proposed FPGA-based accelerator was designed using OpenCL and implemented using the FPGA card DE10-Pro, which contained a Stratix 10 SX FPGA (1SX280HU2F50E1VG) and 8 GB DRAMs. To verify and estimate its performance, sound propagation in a three-dimensional shoebox with dimension being 16m×8m×8m was analyzed. The incidence was an impulse and the number of the computed time steps was 32. As comparison, the counterpart system was developed using C++, and executed on a desktop machine with 512 GB DRAMs and an Intel Xeon Gold 6212U 24-core processor running at 2.4 GHz. The OpenCL codes were compiled using the Intel FPGA SDK for OpenCL 19.1 while the reference C++ codes were compiled using the GNU compiler (version: 4.8.5) with the option -O3 and -fopenmp to use all processor cores.

The average rendering time at each time step is presented in Table I. Compared with the software simulation, the FPGA-based system accelerates computations by 11, 13 and 18 times in second-order, 4th-order, and 6th-order, respectively.

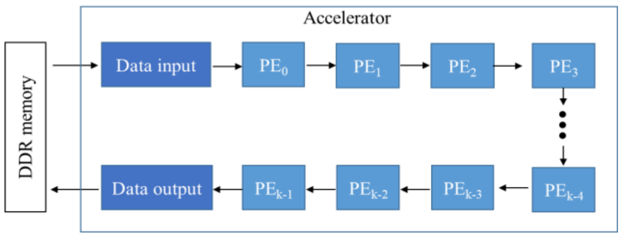

Fig. 1 System diagram

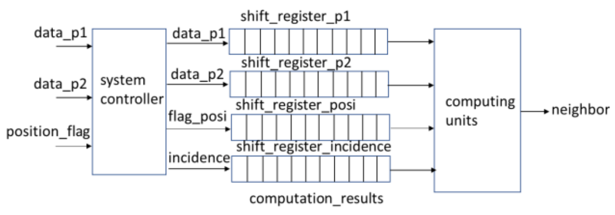

Fig. 2 Structure of a PE

Table I. Rendering time per time step (s)

Orders	FPGA	Software
2nd	0.04859375	0.536307
4th	0.03325	0.445838
6th	0.02384375	0.443655

Acknowledgment
This work was supported by the JSPS KAKENHI Grant Number JP19K12092.

References

1. K. Kowalczyk and M. van Walstijn: IEEE Trans. Audio Speech Lang. Process. 19 [1] (2011) 34-46.
2. J. Mourik and D. Murphy: IEEE/ACM Trans. Audio Speech Lang. Process. 22 [12] (2014) 2003-2011.
3. B. Hamilton and S. Bilbao: Proc. Int. Conference on DAFx, 2013.
4. B. Hamilton and B. Stefan: IEEE/ACM Trans. Audio Speech Lang. Process. 25 (2017) 2112-2124.
5. T. Yiyu, Y. Inoguchi, M. Otani, et al: Appl. Sci. 8 [35] (2018).
6. Y. Tan and T. Imamura: IEEE Cluster, 2020.
7. T. Ishii, T. Tsuchiya, and K. Okubo: Jpn. J. Appl. Phys. 52 (2013) 07HC11.
8. P. Deuflhard and A. Hohmann: Numerical Analysis in Modern Scientific Computing (Springer-Verlag, New York, 2003) 2nd ed.
9. Y. Tan and T. Imamura: USE2019.