ANALYSIS OF SOUND FIELD DISTRIBUTION FOR ROOM ACOUSTICS: FROM THE POINT OF VIEW OF HARDWARE IMPLEMENTATION

Tan Yiyu, Yasushi Inoguchi, Yukinori Sato

Research Center for Advanced Computing Infrastructure, Japan Institute of Science & Technology, Ishikawa, Japan
yiyu-t@jaist.ac.jp
inoguchi@jaist.ac.jp

Hiroshi Matsuoka

Research Institute of Electrical Communication, Tohoku University
Sendai, Japan
matsuoka@riec.tohoku.ac.jp

Makoto Otani

Faculty of Engineering, Shinshu University
Nagano, Japan
otani@cs.shinshu-u.ac.jp

Yukio Iwaya

Faculty of Engineering, Tohoku Gakuin University

Takao Tsuchiya

Department of Information Systems Design
Doshisha University
Kyoto, Japan
ttsuchiy@mail.doshisha.ac.jp

ABSTRACT

Analysis of sound field distribution is a data-intense and memory-intense application. To speed up calculation, an alternative solution is to implement the analysis algorithms by FPGA. This paper presents the related issues for FPGA based sound field analysis system from the point of view of hardware implementation. Compared with other algorithms, the OCTA-FDTD algorithm consumes 49 slices in FPGA, and the system updates 536.2 million elements per second. In system architecture, the system based on the parallel architecture benefits from fast computation since the sound pressures of all elements are obtained and updated at a clock cycle. But it consumes more hardware resources, and a small sound space is simulated by a FPGA chip. In contrast, the system based on the time-sharing architecture extends the simulated sound area by expense of computation speed since the sound pressures are calculated element by element.

1. INTRODUCTION

Modeling the acoustical behavior accurately in a room is complicated because it is affected by the dimensions of a room and the boundary properties. To address it, many methods have been proposed and developed for computer simulation, such as geometrical analysis based acoustical ray tracing [1], and beam tracing method [2], and wave equation based Finite Element Method (FEM) [3], Boundary Element Method (BEM) [4], and Finite Difference Time-Domain (FDTD) method [5]. Moreover, some physical equivalent methods were derived, such as transmission line matrix [6], and digital waveguide mesh (DWM) [7]. Although these methods have advantages in some applications, their common disadvantages are intense computation and high memory requirement as a sound space increases so that the simulations will take a long time by using current computer systems, even if they are much faster.

Analysis of sound field distribution is data-oriented. To calculate the sound pressure of an element, the sound pressures of its neighbors and its own at previous time steps are needed. During calculation, amounts of data are read and written into the memory system, thus in general-purpose computers, memory systems do not work effectively due to the intensive data requirements, lots of data misses in cache, and limited memory bandwidth. To solve this problem, graphics processor units (GPUs) [8] or FPGAs are applied to speed up such type of data-oriented applications [9][12]. In GPU, streaming multiprocessors work in parallel to speed up computation, and data are exchanged by the on-chip shared memory in each streaming multiprocessor rather than the traditional cache or external memory.

FPGA provides another way to analyze the sound field distribution. In FPGA based solution, the wave equations are implemented by programmable logic cells directly, and the temporary data are stored by D flip-flops or block memories. In our previous work, a sound field analysis system based on the two-dimensinal Digital Huygens's Model (DHM), and theoretical analysis for three-dimensional applications were investigated [9][11-13]. In this paper, from the point of view of hardware implementation, design issues about the FPGA based sound field analysis system are discussed. The rest of this paper is organized as follows. Section 2 will discuss the design issues in the FPGA based sound field analysis system. Section 3 will introduce some algorithms for three-dimensional sound field analysis briefly. Section 4 will presents the performance comparison. Finally, conclusions are drawn in Section 5.

2. DESIGN ISSUES IN THE FPGA BASED SOUND FIELD ANALYSIS SYSTEM

In order to achieve high system performance in the FPGA based sound field analysis system, some design issues are considered, such as the algorithm, data dependency and system architecture.

(1) Algorithm. The algorithm determines the architecture of a computing cell. In hardware implementation, complex arithmetic operations consume more hardware resources, and decrease the system clock frequency due to long route delay. Generally, additions, subtractions, and shift operations are implemented by hardware easily and require small hardware resources, while multiplication and division operations are complicate and consume more hardware resources. Thus an algorithm with no or few multiplication and division operations involved is suitable for hardware implemention.

(2) Data dependency. One of the advantages of hardware implementation is to realize the analysis system in parallel to

speed up calculation. However, if data dependency exists, calculations can not be carried out in parallel, instead it occurs in series. So during implementation, data flow in the system needs to be analyzed carefully.

(3) System architecture. System complexity and memory requirement depend on the system architecture, which is another key factor to affect system performance.

2.1. Algorithms for three-dimensional sound field analysis

Many algorithms have been proposed to analyze the sound field distribution. Among them, the FDTD method and its extensions are applied widely, where the wave equations are discretized by the central differential method on staggered grids. Recently, some algorithms were proposed for hardware implementation, such as Yee-FDTD, DHM, and compact explicit FDTD.

2.1.1 Yee-FDTD algorithm

In the Yee-FDTD algorithm [9], the propagation of a sound wave is governed by equation (1).

$$\begin{aligned}\frac{\partial P}{\partial t}+\rho c^2\nabla\bullet u=0\\ \frac{\partial u}{\partial t}+\frac{1}{\rho}\nabla\bullet P=0\end{aligned} \tag{1}$$

Where P is the sound pressure, t is time, ρ is medium density, u is particle velocity, and c is sound wave velocity. When equation (1) is discretized by the central differentiate method, and the assumption $\Delta x=\Delta y=\Delta z=\Delta l$ is applied, equations (2) and (3) are derived to calculate the sound pressure of an element.

$$P^n(\mathrm{i,j,k})=P^{n-1}(\mathrm{i,j,k})-\chi^2\begin{bmatrix}\bar{u}_x^{\,n-\frac{1}{2}}\left(i+\frac{1}{2},j,k\right)-\bar{u}_x^{\,n-\frac{1}{2}}\left(i-\frac{1}{2},j,k\right)\\ +\bar{u}_y^{\,n-\frac{1}{2}}\left(i,j+\frac{1}{2},k\right)-\bar{u}_y^{\,n-\frac{1}{2}}\left(i,j-\frac{1}{2},k\right)\\ +\bar{u}_z^{\,n-\frac{1}{2}}\left(i,j,k+\frac{1}{2}\right)-\bar{u}_z^{\,n-\frac{1}{2}}\left(i,j,k-\frac{1}{2}\right)\end{bmatrix} \tag{2}$$

$$\bar{u}_x^{\,n+\frac{1}{2}}\left(i+\frac{1}{2},j,k\right)=\bar{u}_x^{\,n-\frac{1}{2}}\left(i+\frac{1}{2},j,k\right)-\left(P^n(\mathrm{i+1,j,k})-P^n(\mathrm{i,j,k})\right) \tag{3}$$

Where Δl is the grid size, $\bar{u}_x$, $\bar{u}_y$, $\bar{u}_z$ are the virtual particle velocity at the center point between two neighbor elements, and $\chi=c\Delta t/\Delta l$ is the courant number. If χ is $1/2$, the multiplication operation in equation (2) is replaced by a right shift operation. From equations (2) and (3), 13 operations, including five additions, seven subtractions, and one right-shift operation, are required to calculate the sound pressure of an element. The advantage of the Yee-FDTD algorithm is without multiplication operations invovled. However, data dependency exists. For example, the sound pressure $P^n(i,j,k)$ must be calculated before the calculation for $\bar{u}_x^{\,n+\frac{1}{2}}\left(i+\frac{1}{2},j,k\right)$ is carried out.

2.1.2 Compact Explicit FDTD

A general formulation of the compact explicit FDTD scheme was presented by K. Kowalczyk [10], and the related parameters were given for different stencil grids. Among them, the algorithm for the octahedral grid, namely OCTA-FDTD, is easily implemented by hardware because it is simple and without multiplication operation. Although the algorithms for other stencils can be also implemented by hardware, they are more complex than the OCTA-FDTD algorithm shown in equation (4).

$$\begin{aligned}P^{n+1}(i,j,k)=\frac{1}{4}(&P^n(i+1,j+1,k+1)+P^n(i+1,j-1,k+1)\\&+P^n(i+1,j+1,k-1)+P^n(i+1,j-1,k-1)+P^n(i-1,j+1,k+1)\\&+P^n(i-1,j-1,k+1)+P^n(i-1,j+1,k-1)+P^n(i-1,j-1,k-1))\\&-P^{n-1}(i,j,k)\end{aligned} \tag{4}$$

In equation (4), nine operations, including seven additions, one subtraction, and one right shift operation, are required to calculate the sound pressure of an element. Especially, no data dependency problem exists during computation.

2.1.3 DHM

In the DHM [11][12], a sound space was divided into small acoustic tubes. When a sound pulse is incident into the crossed junctions of acoustic tubes, scatterings occurred because of the impedance discontinuity. Some parts of the incidence are transmitted to the neighbor tubes in six directions, and the rest is reflected back along the incident direction. The relations between the scatterings and incidences are shown in equation (5).

$$\begin{bmatrix}S_1^n\\S_2^n\\S_3^n\\S_4^n\\S_5^n\\S_6^n\end{bmatrix}=\frac{1}{3}\begin{bmatrix}-2&1&1&1&1&1\\1&-2&1&1&1&1\\1&1&-2&1&1&1\\1&1&1&-2&1&1\\1&1&1&1&-2&1\\1&1&1&1&1&-2\end{bmatrix}\begin{bmatrix}P_1^n\\P_2^n\\P_3^n\\P_4^n\\P_5^n\\P_6^n\end{bmatrix} \tag{5}$$

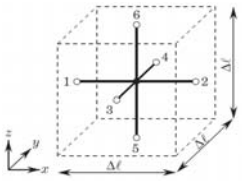


Fig. 1 An three-dimensional DHM element

For an element at the position (*i, j, k*), the relations between S_m^n and P_m^{n+1} (m =*1, 2, … 6*) are shown as follows:

$$\begin{aligned}&P_1^{n+1}(i,j,k)=S_2^n(i-1,j,k) \quad P_2^{n+1}(i,j,k)=S_1^n(i+1,j,k)\\&P_3^{n+1}(i,j,k)=S_4^n(i,j-1,k) \quad P_4^{n+1}(i,j,k)=S_3^n(i,j+1,k)\\&P_5^{n+1}(i,j,k)=S_6^n(i,j,k-1) \quad P_6^{n+1}(i,j,k)=S_5^n(i,j,k+1)\end{aligned} \tag{6}$$

According to equations (5) and (6), the sound pressure of an element is calculated through the incidences and scatterings. The calculation requires twelve operations, including five additions, six subtractions, and one multiplication. This is the original DHM algorithm. To speed up calculation, the updated DHM algorithm shown in equation (7) is derived by inserting equations (5) and (6) into the sound pressure formula and eliminating the pulses $P_m^{n-1}(i,j,k)$. Equation (7) is same as the FDTD expression in the case of SLF stencil scheme (SLF-FDTD)[10].

$$\begin{aligned}P^n(i,j,k)=\frac{1}{3}(&P^{n-1}(i+1,j,k)+P^{n-1}(i-1,j,k)+P^{n-1}(i,j+1,k)\\&+P^{n-1}(i,j-1,k)+P^{n-1}(i,j,k+1)+P^{n-1}(i,j,k-1))-P^{n-2}(i,j,k)\end{aligned} \tag{7}$$

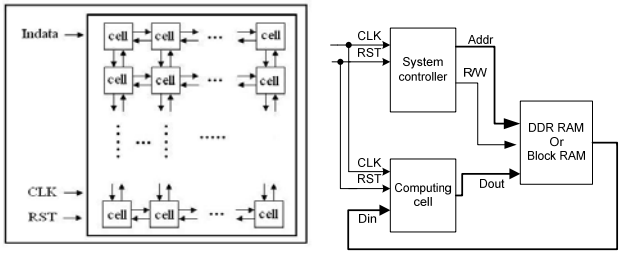
In our previous work [11][12], a two-dimensional DHM algorithm was proposed and implemented by FPGA, where the division by two is replaced by one-bit right shift operation. But in equation (7), the division by three can not be eliminated, thus seven operations are required to calculate the sound pressure of an element, including five additions, one subtraction, and one

multiplication. The advantage of the updated DHM algorithm or SLF-FDTD is only seven operations are required, but the multiplication operation can not be removed.

In principle, the multiplication operations are eliminated by choosing the suitable courant number in the Yee-FDTD algorithm while they are removed through modifying the forms of the grid stencils in the OCTA-FDTD algorithm. In the DHM, the multiplication operations can not be replaced by other simple operations, such as shift operations.

2.2. System architecture

The parallel architecture and time-sharing architecture are applied in the FPGA based sound field analysis system [13]. In the parallel architecture, a computing cell is located at each element to calculate its sound pressure. Thus a sound space is mapped into a hardware based array. For a two-dimensional sound space, the system diagram is shown in Fig. 2a [13].



(a) Parallel architecture (b)Time-sharing architecture

Fig. 2 Diagram of system

In Fig. 2a, a computing cell is designed based on the wave equations and exchanges data with its neighbors. The calculation results are held by D flip-flops inside a FPGA chip. The number of computing cells is determined by the dimensions of a sound space and the grid size Δl. Hence the structure of the computing cell has a great impact on the system performance and hardware resource consumption. Typically, the chosen algorithm and the design techniques will affect the design of a computing cell. The chosen algorithm is the principle factor while circuit design techniques are complements.

The system shown in Fig. 2a is suitable to be implemented by FPGA. A FPGA chip consists of Configurable Logic Blocks (CLBs), which are arranged like an array. The CLBs are connected to each other through the programmable routing matrix, and they are the main logic resources for implementing sequential as well as combinational circuits. Thus a computing cell in the shown architecture can be implemented by one or more CLBs, and they are cascaded together easily through the programmable inter-connectors.

The main problem in the parallel architecture is the data communication. As shown in Fig. 2a, all uniform computing cells are cascaded together and the computation result of a cell is as the inputs of its neighbor cells at the next time step. A common technique is to use handshaking signal between computing cells for data communication. When a computing cell completes the calculation, its output is updated and the handshaking signal is set, then the neighbor cells start to read the updated results and carry out computation. Although the data throughput and timing can be improved in this solution, the calculation efficiency is low. If the output of a computing cell can not be updated on time, all calculations in other computing cells will be suspended, which will result in the overhead increasing. To solve this problem, when a computing cell is designed, all operations in it are limited to be finished in a clock cycle. Hence all computing cells complete computations and updated their outputs synchronously. At next clock cycle, they read data from their neighbors, and no handshaking signals are required. Compared with other solutions, this solution maybe has relatively worse timing performance, the hardware system is simple, and computing efficiency is enhanced.

Another problem in the parallel architecture is hardware resource consumption. Since a computing cell locates at each element, when a sound space becomes larger, the hardware resources will increase rapidly. Thus a sound field analysis system with limit computing cells may be implemented by a FPGA chip. Therefore, a sound field analysis system based on the parallel architecture and implemented by a FPGA chip can only analyze the sound field distribution in a small area.

To extend the simulated area, another architectural solution is to apply the time-sharing architecture, which is shown in Fig. 2b. In Fig. 2b, the sound pressures are calculated element by element through the computing cell module. At a time step, to calculate the sound pressure of an element, the system controller generates the related addresses, then the computing cell accesses the memory to read data according to the addresses, and calculate the sound pressure, finally, the calculation result is written into the memory. This procedure is carried out again until the sound pressures of all elements are obtained. Then a new incidence is read and computations start for the next time step. In Fig. 2b, the memory can be external DDR RAM or block RAMs inside a FPGA chip. Typically, DDR RAM has large capacity, but it can not be accessed in parallel, and only one datum can be read out at one time. The block RAMs inside a FPGA chip can be configured to work in multi-ports mode to read data out in parallel in order to reduce the overhead of data accessing.

The system based on the time-sharing architecture extends the simulated area deeply. For example, when the analysis algorithm is two-dimensional DHM, the system capacity based on the time-sharing architecture is improved about 20 times in a FPGA chip, namely, the simulated area is increased 20 times [13]. However, its computation performance will be decreased 20 times. In the parallel architecture, the sound pressures of all elements are obtained at one clock cycle, while they are retrieved element by element in the time-sharing architecture. Thus the computation time spent in the time-sharing architecture is 20 times as that taken in the parallel architecture. This problem can be sovled by system partition methods, where a system is divided into several small systems, and they work in parallel.

3. PERFROMANCE COMPARISON

3.1. Performance estimation

Table 1 shows the performance estimation of different algorithms. In Table 1, the number of slices denotes the hardare resource consumption in the Xilinx FPGA XC5VLX330T, and the block RAMs RAMB18X2s and RAMB36_EXPs are used to implement the multiplication operations. The maximum frequency is the maximum clock frequency system works after synthesis. Although the Yee-FDTD algorithm does not require multiplication operation, it consumes more hardware resources due to more arithmetic operations, and the data dependency results in the decrease of the maximum clock frequency. Since the DHM algorithms and SLF-FDTD needs multiplication operations, some extra block RAMs are needed. The elements updated per second presents the system execution performance based on the assumption all operatoins are finished in a clock cycle in a computing cell. From the Table, the OCTA-FDTD algorithm consumes least

hardware resources, and has best execution performance due to its simplicity and no multiplication involved.

Table 1 Performance estimation of different algorithms

Algorithm	No. of Slices	RAMB 18X2s	RAMB 36_EXPs	Cells updated per second (millions)
Yee-FDTD	142	0	0	73.9
DHM(original)	118	2	3	140.7
DHM(updated) or SLF-FDTD	81	1	3	153.7
OCTA-FDTD	49	0	0	536.2

3.2. Memory requirement

Sound field analysis is a memory-intensive application. Inside a FPGA chip, data are stored by D flip-flops or block memory. Compared with the solution to storing data in the external memory, the overhead to read data from D Flip-flops or block memory is much shorter. Moreover, the bottleneck of performance improvement due to the bandwidth limitation of external memory can be eliminated. If a sound space is divided into $N\times M\times K$ elements and data are 32-bit, from equations (2) and (3), in the Yee-FDTD algorithm, at each element, except the sound pressure at previous time step is stored, the virtual particle velocity at x, y, z directions is also needed to be kept for further calculation. Hence at each element, four data are stored, and the system needs *16NMK* byte memory.

From equation (4), the sound pressures of an element at previous one and two time steps are kept in the OCTA-FDTD algorithm. Thus the system needs *8NMK* byte memory. In the DHM original algorithm, at each element, scatterings and incidences at six directions are stored, thus the system requires *48NMK* byte memory. In contrast, the sound pressures of an elelment at previous one and two time steps are kept in the DHM updated algorithm or the SLF-FDTD, and the system needs *8NMK* byte memory. Among all these algorithms, the OCTA-FDTD and DHM updated algorithms require smallest size memory.

3.3. Calculation error

In FPGA implementation, calculation error mainly results from data truncation error and arithmetic overflow. Since data are integer in hardware system in order to simplify system design, truncation error will appear. It can be reduced by increasing data width or changing data representation format. But the system complexity and hardware resources consumption will increase with data width increasing. On the other hand, arithmetic overflow introduces errors, such as addition overflow. Sometimes this type of error is accumulated. Thus the more arithmetic operations are carried out, the larger errors will be introduced. Typically, they are avoided by data width extension. More arithmetic operations require more data bits extension, which will results in more hardware resource consumption. In addition, dispersion errors exist in the FDTD scheme. Compared with the SLF-FDTD, the OCTA-FDTD algorithm has smaller dispersion error.

4. CONLCUSION

Sound field analysis is a data-oriented and intense memory requirement application. FPGA devices provide another solution to it through direct hardware implementation of propagation equations of sound wave. In hardware implementation, in order to reduce hardware resource consumption and improve system performance, multiplication operations are required to be reduced or eliminated. Based on this, the OCTA-FDTD algorithm is more suitable for hardware implementation, which consumes small hardware resources, and has no multiplicaton operations involved. From the point of view of architecture, the parallel architecture provides faster computation, but consumes more hardware resources. In contrast, the time-sharing architecture extends simulated area by degrading low computation speed.

5. ACKNOWLEDGMENTS

This project is supported by the National Institute of Information and Communication Technology.

6. REFERENCES

[1] A. Krokstad, S. Strom, and S. Sorsdal, “Calculating the acoustical room response by the use of a ray tracing technique,” *Journal of Sound Vibration*, vol. 8, no. 1 pp. 118–125, 1968.

[2] T. Funkhouser, N. Tsingos, I. Carlbom, G. Elko, et al, “A beam tracing method for interactive architectural acoustics,” *Journal of the Acoustical Society of America*, vol. 115, pp.739–756, 2004.

[3] J. N. Reddy, *An Introduction to the Finite Element Method*, 3rd ed., McGraw-Hill, USA, 2006.

[4] P. K. Banerjee, and R. Butterfield, *The Boundary Element Methods in Engineering Science,* McGraw-Hill, USA, 1994.

[5] D. Botteldooren, “Finite-difference time-domain simulation of low-frequency room acoustic problems,” *Journal of the Acoustical Society of America*, vol. 98, no. 6, pp. 3302–3308. 1995.

[6] P. B. Johns, R. L. Beurle, “Numerical solution of 2 dimensional scattering problems using a transmission-line matrix,” *Proceedings of IEE* 118 (9) (1971) 1203-1208.

[7] G. R. Campos and D. M. Howard, “On the computational efficiency of different waveguide mesh topologies for room acoustic simulation,” *IEEE Transactions on Speech and Audio Processing,* vol. 13, no. 5, pp.1063-1072, 2005.

[8] L. Savioja, Real-time 3D finite-difference time-domain simulation of low- and mid-frequency room acoustics, *13th International Conference on Digital Audio Effects*, *DAFx-10*, September 2010.

[9] Yasushi Inoguchi, Tan Yiyu, Yukinori Sato, et al, “DHM and FDTD based hardware sound field simulation acceleration,” *14th International Conference on Digital Audio Effects, DAFx-11*, pp. 69-72, 2011.

[10] Konrad Kowalczyk, and maarten van Walstijn, “Room Acoustics Simulation Using 3-D Compact Explicit FDTD Schemes,” *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 19, No. 1, pp.34-46, 2011.

[11] Tan Yiyu, Yasushi Inoguchi, Eiko Sugawara, Yukinori Sato, et al, “A FPGA Implementation of the Two-Dimensional Digital Huygens' Model,” *2010 International Conference on Field Programmable Technology (FPT),* pp. 304 – 307.

[12] Tan Yiyu, Yukinori Sato, Eiko Sugawara, Yasushi Inoguchi, et al, “A real-time sound field renderer based on digital Huygens' model,” *Journal of Sound and Vibration,* vol. 330, pp. 4302-4312, 2011.

[13] Tan Yiyu, Yasushi Inoguchi, Yukinori Sato, et al, “Design of a FPGA-based Timing Sharing Architecture for Sound Rendering Applications,” *International Conference on Information Technology: Next Generations*, USA, April, 2012.