3Pa2-2

Proceedings of Symposium on Ultrasonic Electronics, Vol. 33 (2012) pp. 321-322
13-15 November, 2012

A Hardware-oriented FDTD Algorithm for Sound Field Rendering

Tan Yiyu[1†], Yasushi Inoguchi, Yukinori Sato [1], Makoto Otani [2],
Yukio Iwaya[3], Hiroshi Matsuoka[4], Takao Tsuchiya[5]
[1] Res. Center for Adv. Comp. Infra., Japan Adv. Inst. of Sci. & Tech.; [2] Faculty of Eng., Shinshu Univ.; [3] Faculty of Eng., Tohoku Gakuin Univ.; [4]Faculty of Eng., Tohoku Univ.; [5]Dept. of Info. Syst. Design, Doshisha Univ.

1. Introduction

Sound field rendering is becoming more and more important in some applications. Many algorithms based on computer simulation have been proposed for sound field rendering, such as acoustical ray tracing[1)], image source method[2)], beam tracing method[3)], finite element method[4)], boundary element method[5)], and finite difference time-domain (FDTD)[6)]. Although these algorithms are well defined and accurate for certain applications, their common drawbacks are the intense computations and the high memory requirements as a sound space increases. Due to the limited memory bandwidth of current general-purpose computer systems, such simulations will take a long time. An alternative solution is to implement the sound field rendering algorithms directly by hardware. In our previous work, the Digital Huygens' Model was proposed for the Field Programmable Gate Array (FPGA) based two dimensional sound field rendering[7,8)]. In this paper, the RMWE-FDTD algorithm is introduced for three dimensional sound field rendering.

In the hardware-based sound field rendering system, a sound space is divided into lots of uniform cubic elements. A uniform computing cell designed based on the rendering algorithm is located in each element to calculate the sound pressure. Typically, a computing cell is required to match the following conditions to improve the system performance.

(1) The computing cell runs at as fast as possible to reduce the calculation time.
(2) The computing cell consumes small hardware resources.

Based on these requirements, the sound field rendering algorithm must be as simple as possible, i.e. without complex arithmetic operations involved, such as multiplication and division, since these operations consume more hardware sources, and degrade system clock due to long route delay.

2. RMWE-FDTD Algorithm

In a cubic element, sound wave propagation is governed by the following formula:

$$\frac{\partial^2 p}{\partial t^2} = c^2 (\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} + \frac{\partial^2 p}{\partial z^2}) \qquad (1)$$

where p denotes the sound pressure, c is the sound speed, x, y, and z are directions of an $x-y-z$ Cartesian coordinate system.

By applying the center differential method in equation (1), and let $\Delta x = \Delta y = \Delta z = \Delta l$, then

$$\begin{aligned} P^{n+1}(i,j,k) &= 2P^n(i,j,k) - P^{n-1}(i,j,k) + \chi^2[P^n(i+1,j,k) \\ &+P^n(i-1,j,k) + P^n(i,j+1,k) + P^n(i,j-1,k) \\ &+P^n(i,j,k+1) + P^n(i,j,k-1) - 6P^n(i,j,k)] \end{aligned} \qquad (2)$$

where $\chi = c\Delta t/\Delta l$ denotes the Courant number. For a three dimensional sound space, $\chi \le 1/\sqrt{3}$. When χ is $1/\sqrt{3}$, equation (2) is the same as the standard leapfrog Yee-FDTD[12)] and the updated DHM[10)].

To eliminate the multiplication operation, χ is assumed to be $1/2$, then equation (2) is changed as.

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

In equation (3), the multiplication operations 2^m (m= -2, 1) are implemented by shift operations. In contrast with equation (2), equation (3) consists of six additions, one subtraction, and two shift operations.

3. System Performance

3.1 Memory requirement

From equation (3), the sound pressures of elements in the current time step and the previous one time step are stored during calculation. If the number of elements is N, and data width is 32-bit, the memory requirement, the number of operations and memory access in different algorithms are shown in **Table I**. Except the sound pressures of elements in the current time step are kept, the particle velocity at x, y, and z directions are stored in the algorithm Yee-FDTD[9,13)], and the scatterings at six directions are required in the original DHM[10,11)]. This results in more memory and operation requirements for the Yee-FDTD and

----

yiyu-t@jaist.ac.jp inoguchi@jaist.ac.jp

original DHM algorithms[13]. Although the number of operations is the least in the updated DHM, a multiplication operation is included. Compared with other algorithms, the RMWE-FDTD algorithm needs smallest memory, relatively less memory access and operations, and has no multiplication involved.

Table I Memory requirement, operations, and memory access in different algorithms

Algorithm	Memory (byte)	Operations		Memory access
		Total	Multiplication	
Yee -FDTD	16N	13	0	18
Original DHM	48N	12	1	13
Updated DHM	8N	7	1	8
REWE-FDTD	8N	9	0	9

3.2 Computational results

To verify the validity of the proposed algorithm, a sound space with 32768 (32 × 32 × 32) elements surrounded by rigid walls is examined and a related sound field rendering system based on the proposed algorithm is developed by both software and hardware. In the sound space, an incident source is located at the point (16, 16, 16), and the observation point is at the point (16, 16, 14). The incident source is a single-shot sine wave with amplitude of 10^8 Pa and 16 samples taken at each period. The calculated time steps are 160. The system is simulated by both software and hardware. The software simulation is developed using the C++ programming language, and the hardware system is synthesized in a FPGA chip Xilinx XC5VLX330T-FF1738, and simulated by the EDA tool called Modelsim. In the software simulation, the data are integers while they are 32-bit fixed point in the hardware simulation. **Fig. 1** shows the sound pressure at the observation point in both the hardware and software simulations. Except for the one cycle delay, the hardware simulation results agree well with those of the software simulation.

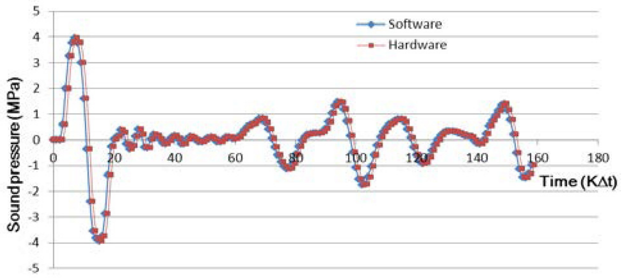

Fig. 1 Wave at the observation point

3.3 Calculation time

In order to evaluate the calculation performance and make a comparison, systems based on different rendering algorithms are developed by C++. In these systems, data are 32-bit floating-point. And the sound space, incident signal, and observation point, are same as the previous ones except the calculation time steps are 20,000. The simulation environment is a computer with an AMD Phenom 9500 Quad-core processor (CPU frequency: 2.2 GHz) and 4Gb RAM. The calculation time taken by different algorithms is presented in **Table II**.

In Table II, the courant number χ is $1/\sqrt{3}$ in the updated DHM and Yee-FDTD, thus their rendering equations are the same, and systems take the same time. The RMWE-FDTD speeds up about 19% (11.95/10.01-1) against the updated DHM and Yee-FDTD, and about 132% (23.23/10.01-1) against the original DHM in compuations, respectively. This performance enhancement results from no complicated multiplication operations involved in the RMWE-FDTD.

Table II Execution time taken by different algorithms (s)

Element Scale	RMWE-FDTD	Original DHM	Updated DHM	Yee-FDTD
32 × 32 × 32	10.01	23.23	11.95	11.95

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

References

1. A. Krokstad, S. Strom, and S. Sorsdal: J. Sound Vib. 8[1] (1968) 118.
2. J. B. Allen and D. A. Berkley: Journal of the Acoustical Society of America, 65[4] (1979) 943.
3. T. Funkhouser, N. Tsingos, I. Carlbom, G. Elko: J. Acoust. Soc. Am. 115 (2004) 739.
4. J. N. Reddy: An Introduction to the Finite Element Method (McGraw-Hill, USA, 2006).
5. P. K. Banerjee, and R. Butterfield: The Boundary Element Methods in Engineering Science (McGraw-Hill, USA, 1994).
6. D. Botteldooren: J. Acoust. Soc. Am. 95 [5] (1994) 2313.
7. T. Yiyu, Y. Inoguchi, E. Sugawara, et al: J. Sound Vib. 330 (2011) 4302.
8. T. Yiyu, Y. Inoguchi, E. Sugawara, et al: Proc. Int. Conference on Field Programmable Technology (FPT), 2010, p. 304.
9. T. Tsuchiya, T. Ishii, and K. Okubo: IEICE Tech. Rep. 111[88] (2011) 25 [in Japanese].
10. T. Tsuchiya, and Y. Kagawa: Jpn. J. Appl. Phys. 44 [6B] (2005) 4297.
11. T. Tsuchiya: Jpn. J. Appl. Phys. 46 [7B] (2007) 4809.
12. K. Kowalczyk and M. van Walstijn: IEEE Trans. Audio Speech and Language Proces.19[1] (2011) 34.
13. T. Yiyu, Y. Inoguchi, et al: Proc. Int. Conference on DAFx, 2012 (accepted).