

RESEARCH

Open Access

# Simulation of ultrasound nonlinear propagation on GPU using a generalized angular spectrum method

Francois Varray<sup>1,2\*</sup>, Christian Cachard<sup>1</sup>, Alessandro Ramalli<sup>1,2</sup>, Piero Tortoli<sup>2</sup> and Olivier Basset<sup>1</sup>

## Abstract

Acoustic simulation has always played an important role in the development of new ultrasound imaging techniques. In nonlinear ultrasound imaging particularly, the simulators are accurate but time-consuming, because of the high derivative order of the propagation equation and to the classic solution based on finite difference schemes. This article presents a fast 3D +  $t$  nonlinear ultrasound simulator, based on a generalized angular spectrum method, particularly fit for the graphics processing unit (GPU). Indeed, the Fourier domain approach decreases the derivative order of the propagation, thus significantly speeding up the simulation time. The simulator was implemented and optimized on a central processing unit (CPU) and a GPU, respectively. The processing times measured on two different graphic cards show that, compared to the CPU, GPU-based implementation is 3.5-13.6 times faster.

## 1. Introduction

The use of harmonic imaging in ultrasound has become popular because of the improvement it offers in terms of axial and lateral resolution with respect to standard B-mode imaging [1]. The new modality exploits the nonlinear propagation of ultrasound waves in human tissues, yielding the presence of significant harmonics in the ultrasound echoes, which can be selected on the receiver. The great interest in nonlinear propagation and its applications has stimulated the development of simulation programs, capable of predicting the behavior of a large class of ultrasound waves in different tissues. The main strategies to simulate the distortion of a propagating wave are based on the finite difference approach [2] and the angular spectrum method (ASM) [3]. The former is more accurate, but requires a very long time to converge. On the other hand, the angular approach is faster because it solves the propagation equation in the Fourier domain, thus decreasing the derivative terms of the propagation equation. Furthermore, by considering the fundamental and the second harmonic distortion separately, simpler and faster

solutions [4-7] can be obtained. In the recent article by Wojcik et al. [8], the simulation time for a 4D (3D + time) ultrasound wave propagation was between 3 and 12 h. Although an ASM-based simulator decreases the computation time, for a 3D +  $t$  volume, it still takes about 2 h [9]. Work is required to optimize the ASM approach and to develop a fast simulation tool.

In the last few years, the increased performance of graphics processor units (GPUs) has made them excellent candidates not only for display but also for intensive calculus, and different applications have been transferred from central processing units (CPUs) to GPUs. The increasing number of cores on a GPU can be exploited for high-level parallelism and intensive simulations. Recent works in ultrasound demonstrate the potential of the GPU in several applications such as, e.g., Doppler imaging [10], block-matching [11], synthetic aperture technique [12], or volume rendering [13,14]. In terms of ultrasound image simulations, Kutter et al. [15] proposed a CT-image-based method using a linear convolution model. The resulting images are realistic, taking into account both the acoustic impedance variations and the shadow effects thanks to the CT images, but did not correctly consider the beamforming issue in transmission and reception. The simulation of a complete beamforming strategies is time-

\* Correspondence: francois.varray@creatis.univ-lyon1.fr

<sup>1</sup>Université de Lyon, CREATIS; CNRS UMR5220; INSERM U1044; INSA-Lyon; Université Lyon 1; 7 av Jean Capelle, 69621 Villeurbanne, France  
Full list of author information is available at the end of the article

consuming, and Shams et al. [16] tried to solve this issue by computing the spatial impulse response of the transducer on a GPU. In terms of nonlinear propagation, Pinton et al. [17] proposed a complete finite difference scheme to compute the nonlinear propagation and to consider possible inhomogeneity in the density or speed of sound. Karamalis et al. [18] proposed to solve the Westervelt equation on a GPU through a finite difference calculation. This approach computes the 2D +  $t$  wave propagation, while the image reconstruction is performed on a CPU.

In this article, a GPU implementation of the generalized ASM (GASM) in a 3D +  $t$  configuration is presented. The mathematical and the acoustic background of the GASM have already been presented in [19] and the resulting fields are in good accordance both with the experimental one. The GASM differs from the classic ASM approach since it also considers an inhomogeneous nonlinear parameter in the simulated medium. Currently, no simulation tools based on the ASM have been implemented on a GPU, whereas ASM is the more potential method to access to the fastest nonlinear propagation; thanks to the mathematical background which naturally decrease the computation time compare to finite difference methods. It is shown here that the GPU implementation perfectly fits in the mathematical features of the GASM, yielding a significantly decreased computation time.

The next section reviews the GASM, which can compute the fundamental and second harmonic evolution separately. The second part is dedicated to the GPU implementation of the method, and the different choices made to increase its performance are discussed. The results obtained with this implementation are presented in Section 4, where they are also compared to the results of a classic CPU implementation.

## 2. Angular spectrum method

In a lossless medium, the evolution of the ultrasound pressure in 4D (3D +  $t$ ) can be decomposed into its fundamental ( $p_1$ ) and second harmonic ( $p_2$ ) components and their time and spatial evolution can be expressed as [6,7,20]

$$\left(\Delta - \frac{1}{c_0^2} \frac{\partial^2}{\partial t^2}\right) p_1(z, x, y, t) = 0 \quad (1)$$

$$\left(\Delta - \frac{1}{c_0^2} \frac{\partial^2}{\partial t^2}\right) p_2(z, x, y, t) = -\frac{\beta}{\rho_0 c_0^4} \frac{\partial^2 p_1^2(z, x, y, t)}{\partial t^2} \quad (2)$$

respectively, with  $c_0$  the speed of sound,  $\rho_0$  the density,  $\beta$  the nonlinear parameter, and  $\Delta$  the Laplacian, which takes into account the diffraction phenomenon of the ultrasound probe

$$\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2} \quad (3)$$

In order to decrease the derivative order of (1) and (2), the Fourier transform (FT) of each equation must be computed. The FT ( $F$ ) and the inverse FT (IFT,  $F^{-1}$ ) are, respectively, defined as

$$F(p_j) = P_j(z, f_x, f_y, f_t) \int \int \int p_j(z, x, y, t) e^{-i2\pi(f_x x + f_y y - f_t t)} dx dy dt \quad (4)$$

$$F^{-1}(P_j) = p_j(z, x, y, t) = \int \int \int P_j(z, f_x, f_y, f_t) e^{i2\pi(f_x x + f_y y - f_t t)} df_x df_y df_t \quad (5)$$

with  $j$  equal to 1 or 2 if the computed pressure is related to the fundamental or to the second harmonic component, respectively,  $f_x$  and  $f_y$  are the spatial frequencies in the  $x$  and  $y$  directions, and  $f_t$  the temporal frequency. Considering the well-known properties of the FT

$$F\left(\frac{\partial^n p}{\partial v^n}\right) = (-2i\pi f_v)^n F(p) \quad (6)$$

$$F\left(\frac{\partial^n p}{\partial t^n}\right) = (2i\pi f_t)^n F(p) \quad (7)$$

where  $v$  corresponds to  $x$  or  $y$ , and applying the FT to (1) and (2), the following equations are obtained

$$\frac{d^2 P_1}{dz^2} + K^2 P_1 = 0 \quad (8)$$

$$\frac{\partial^2 P_2}{\partial z^2} + K^2 P_2 = \frac{k_t^2}{\rho_0 c_0^2} F(\beta p_1^2) \quad (9)$$

$$K(k_x, k_y, k_t) = \sqrt{k_t^2 - k_x^2 - k_y^2} \quad (10)$$

where  $k_j$  is the wave number in the  $j$  direction and  $K$  the complex 3D wave number vector, which depends on the different sampling frequencies and is expressed in  $m^{-1}$ . In the computation, only the real part of the  $K$  vector is considered, because the imaginary part corresponds to a negligible evanescent wave [21]. Equations 8 and 9 can be solved in the Fourier domain, so that the spatiotemporal solutions are then obtained by an IFT. Thus, the final expressions for  $p_1$  and  $p_2$  are

$$p_1(z, x, y, t) = F^{-1}\left(P_0(z_0, k_x, k_y, k_t) e^{-iK(z-z_0)}\right) \quad (11)$$

$$p_2(z, x, y, t) = F^{-1}\left(\frac{-if_t^2}{2\rho_0 c_0^2 K} \left(\int_{z_0}^z F(\beta p_1^2) e^{iKu} du\right) e^{-iKz}\right) \quad (12)$$

with  $P_0$  the FT of the source wave  $p_0$  at depth  $z_0$ . It has to be noted that since the nonlinear coefficient  $\beta$

could depend on the  $x$ ,  $y$ , and  $z$  directions it must be kept inside the FT computation. This consideration allows simulating inhomogeneous nonlinear media. If  $\beta$  is considered constant and then get out of the FT, the obtained formulation would be similar to the one proposed in the literature [9].

### 3. CPU/GPU implementation of the GASM

The solution of Equations 11 and 12 is particularly well suited to GPU programming. Indeed, the different calculations are separately performed in the  $x$ ,  $y$ , and  $t$  dimensions. The different pressures can be seen as pressure images or matrices in 3D ( $x$ ,  $y$ , and  $t$ ). Each product and sum in these 3D images involves one voxel at a given position in the image. This type of calculation is very efficient on a GPU because only the current position is used in the different input and output images. To compute the Fourier transform, which is time-consuming, an external library is used. The GASM is implemented on a CPU and a GPU to compare their performance. The reference CPU programming was done in C++ and the GPU programming was done in Compute Unified Device Architecture, which is an extension of the standard C language developed by nVIDIA. This is an application programming interface that is used to create the parallel programming tasks, called kernels, which are executed on the GPU.

#### 3.1 Computation of $P_1$

The evolution of the fundamental component is only linked to the initial wave source,  $P_0$ , and to the propagation distance,  $z$ . In Equation 11, the exponential term corresponds to a complex rotation, and a specific kernel needs to be designed. This kernel will be used several times in the GASM implementation. The  $P_1$  spectrum is obtained after the computation of the rotation kernel in the Fourier domain, and then the IFT is used to obtain the final solution. It must be noted that the fundamental wave component does not depend on the  $z$ -axis sampling used. Indeed, its evolution can directly be computed anywhere in the medium.

#### 3.2 Computation of $P_2$

The second harmonic wave component will be solved in five steps. First, from the initial  $p_1$  image, the new term  $\beta p_1^2$  has to be computed. Second, the FT of the resulting image is done. Third, the spectrum is rotated. Fourth, the spectrum has to be integrated. Finally, the integrated Fourier spectrum must be rotated once more. The different rotations are defined with the same rotation kernel. To compute the  $P_2$  wave, the  $z$  sampling used is important because of the integrative part. The descriptions of the different kernels are highlighted hereafter.

### 3.3 Fourier transform

The FT library used in the CPU implementation is the FFTW library, which is considered the most efficient in the community [22]. Otherwise, for the GPU implementation, nVIDIA proposed a dedicated library, cuFFT, which is an extension of the FFTW library. Defining  $p_1$  and  $p_2$  as 3D real images and  $P_1$  and  $P_2$  as complex means the dimension of the complex image can be halved and also the computation time in both the FT and IFT decreased.

### 3.4 Kernel description

The kernels used in the GPU implementation are described below. The different kernels are particularly suitable for the GPU because the mathematical operations used in the GASM only involved the voxels at a given position in the 3D images. No access to other specific memory areas is needed to compute the output images, which is very efficient in GPU programming.

#### 3.4.1. Rotation kernel

To compute the fundamental and the second harmonic, a rotation kernel is needed. According to the Euler formula, the complex exponential is considered in its Cartesian form, and then a classic multiplication is computed to obtain the new complex number.

#### 3.4.2 Kernel to compute $\beta p_1^2$

Usually, in a biological medium, the nonlinear parameter  $\beta$  is inhomogeneous in all the directions. The related 3D map is saved in the texture memory in order to easily access its values. With this initial setup, the ( $x$ ,  $y$ ,  $z$ ) sampling has no impact on the computation of the product  $\beta(x,y,z)p_1^2(x,y,z,t)$ . Indeed, the bilinear interpolation, naturally present in the texture memory, is used to extract the correct value of the investigated plane. Concerning multiplication, since  $p_1$  is real, its value is simply multiplied by itself to obtain the square value. This operation is very efficient in GPU programming.

#### 3.4.3. Kernel to compute the integral

The integral computation is the most complex part. In order to compute it, a finite difference scheme was used. Contrary to the fundamental evolution computation, a  $z$  sampling is needed and is defined by the finite difference scheme

$$I(z + dz) = \int_{z_0}^{z+dz} M(u)du = I(z) + \frac{M(z) + M(z + dz)}{2} dz \quad (13)$$

To compute the integral at the  $z + dz$  position, two previously computed values have to be saved, i.e., the previous value of the integral  $I(z)$  and the image  $M(z)$ , which take into account the value of the fundamental pressure at a distance  $z$ . In the kernel, the sums and multiplications have to be computed for the  $z + dz$  position and then are saved for the calculation of the next

position. The different constants are also summed in this kernel.

### 3.5 Final algorithm

The final algorithm is described in Table 1. For each  $z$  position, the fundamental and then the second harmonic components are computed.

## 4. Results

### 4.1 Speed increment

Two different CPUs and GPUs were used to estimate the algorithm's performance and are described in Table 2. To obtain the best performance with the FT library, every dimension of the 3D images must be a power of 2. The estimation of the algorithm has been made through the calculation of four working 2D +  $t$  volumes ( $x, y, t$ ):  $64 \times 64 \times 128$ ,  $64 \times 64 \times 256$ ,  $128 \times 64 \times 256$ , and  $128 \times 128 \times 256$ . The computation time measured takes into account the complete execution of the algorithm: memory allocation, memory transfer, execution, and memory flush.

The resulting calculation times are reduced by a factor of  $3.5 \pm 0.2$  on the Quadro NVS 160 M and  $13.6 \pm 2.1$  on the GTX 285. The difference in these ratios is explained by the higher performance of the GTX GPU, which is composed of more cores and larger memory (see Table 2). The evolution of the computation time for the four volumes can be observed in Figure 1. It can be noted that the GPU computation time on machine 1 is also faster than the CPU on machine 2.

Regarding the computation time, it can be noted that an increase of a factor 30 in the number of GPU cores leads to a relatively weak performance gain. However, the processing times on the Quadro NVS and on the GTX GPU are 360 and 47 ms, respectively, for a working 2D +  $t$  volume of dimension  $128 \times 64 \times 256$ . Those times also consider the memory transfer time since it is not possible to allocate directly on the GPU the whole 3D +  $t$  volume. Indeed, the working volume is composed approximately of two million voxels. Adding a dimension in the  $z$ -direction, for example 50 points,

**Table 1 Illustration of the different steps of the GASM**

$p_0 \rightarrow p_0 \rightarrow$ [FT]
For each $z$ point:
$p_0 \rightarrow p_1 \rightarrow$ [rotation kernel]
$p_1 \rightarrow p_1(z) \rightarrow$ [IFT]
Compute $\beta p_1^2 \rightarrow$ [ $\beta p_1^2$ kernel]
$\beta p_1^2 \rightarrow F(\beta p_1^2) \rightarrow$ [FT]
Rotate $F(\beta p_1^2) \rightarrow$ [rotation kernel]
Compute integral $I \rightarrow$ [integral kernel]
$I \rightarrow p_2 \rightarrow$ [rotation kernel]
$p_2 \rightarrow p_2(z) \rightarrow$ [IFT]

The different FTs, IFTs, and kernels are represented in square brackets.

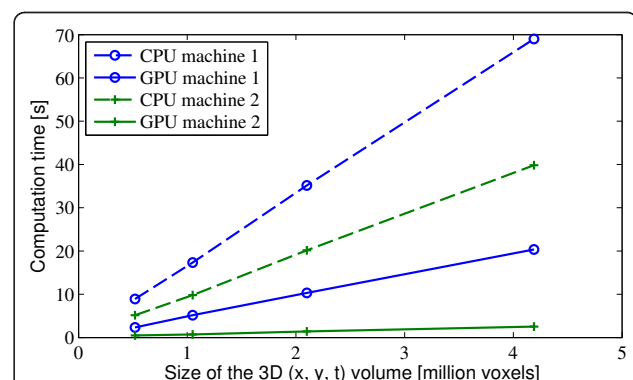
**Table 2 Description of the two CPUs and GPUs used**

	Machine 1	Machine 2
CPU Processor name	Intel Core2 Duo T9400	Intel Xeon E5220
Speed	2.53 GHz	2.27 GHz
Memory	3.48 GB	5.9 GB
GPU Name	Quadro NVS 160M	GTX 285
Global memory	256 MB	1024 MB
Number of multiprocessors	1	30
Number of cores	8	240

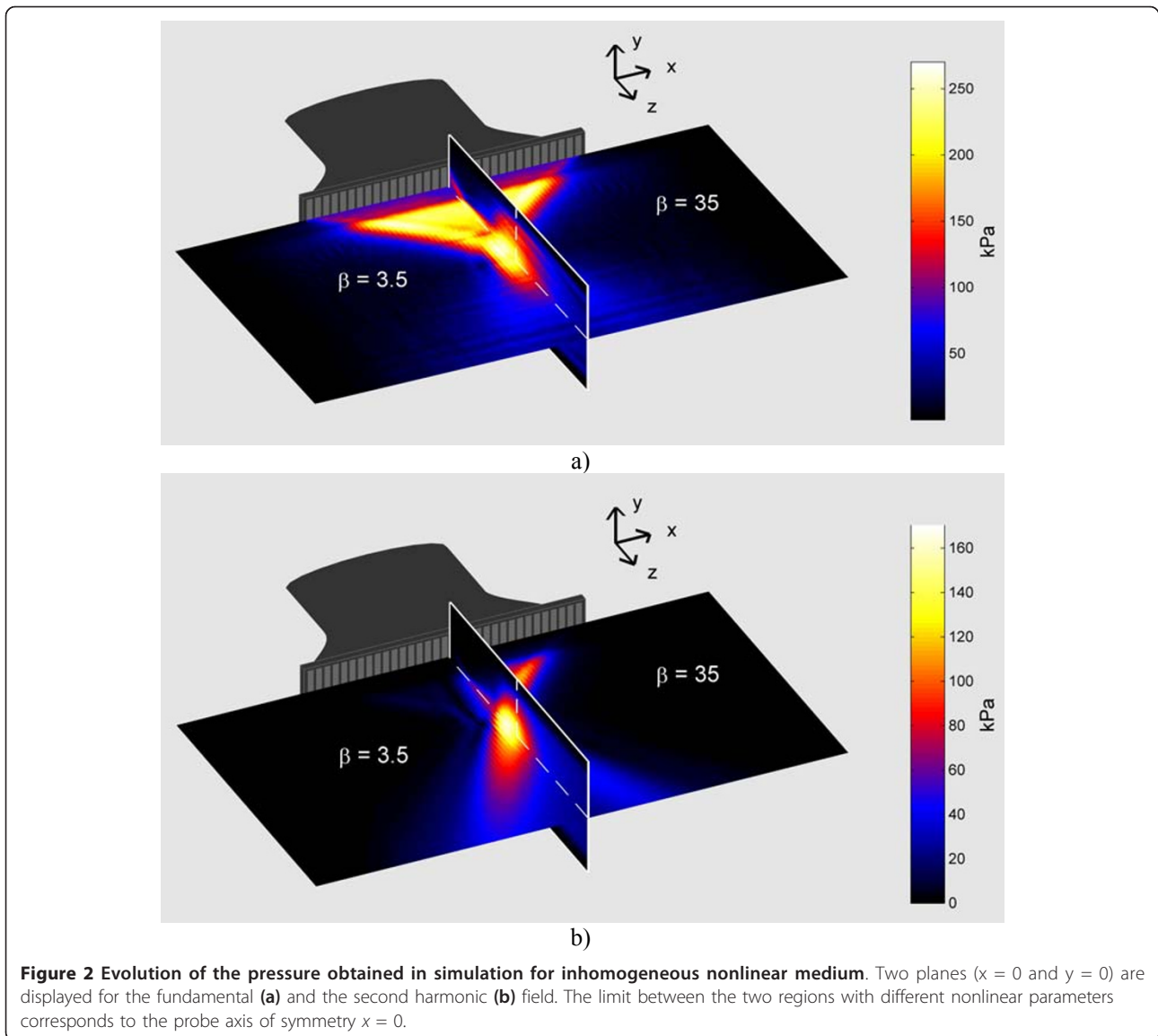
leads to a 3D +  $t$  volume containing more than 100 million voxels. This amount of data has to be considered both for the fundamental and the second-harmonic components. Then, after each  $z$  step, the computed working 2D +  $t$  volume has to be saved on the CPU memory. These memory transfers are time consuming and limit the performance of the proposed method. On both the GPUs, the total memory transfers take around 27 ms, meaning that the performance of the GTX GPU, considering only the execution time is 16.5 better than the Quadro NVS board.

### 4.2 Resulting fields

One possible application of the GASM is to calculate the pressure evolution in a medium with an inhomogeneous nonlinear coefficient. In such cases, the second harmonic pressure is expected to sharply increase according to the nonlinear parameter. For example, Figure 2 shows the ultrasound fundamental and second harmonic fields computed in a plane in which the nonlinear parameter,  $\beta$ , abruptly changed from 3.5 to 35. The peak pressures estimated in two different planes ( $x = 0$  and  $y = 0$ ) are shown according to a color bar. As



**Figure 1 Computation time on the CPU (dotted lines) and GPU (full lines) for the two different PCs.** The curves with 'o' correspond to the laptop (machine 1) and the curves with '+' to a standard PC (machine 2). The total time takes into account for calculating the complete 3D +  $t$  volume, which depends on the  $z$  sampling. The number of points in the  $z$ -axis is 30.



expected, the nonlinear parameter has no impact on the fundamental image (Figure 2a), but the pressure of the second harmonic field significantly increases in the region where  $\beta$  is higher (Figure 2b).

## 5. Discussion and conclusions

Currently available ultrasound simulators, such as FieldII [23], present two weak points: they consider linear propagation only and they need a long time to conduct the simulation. The GPU implementation of the GASM aims at solving both problems. The implementation of the GASM on a GPU significantly decreases the computation time for the fundamental and second harmonic components of an ultrasound field. The GPU is particularly suitable for the GASM because it uses the FFT library designed for parallel computation as well as the kernel that makes calculations

voxel by voxel in a 3D image. The final increase in speed measured on a laptop graphics card (Quadro NVS 160 M) is 3.5. On a more recent and efficient GPU, the speed is 13.6 times faster. The algorithm proposed here could also be extended to higher-order harmonics (third, fourth, etc.) as only one kernel is needed to compute the propagation of each harmonic [24]. However, it has been highlighted that the time devoted to the memory transfers from the GPU to the CPU is not negligible and that increasing the GPU computational power is not sufficient to further decrease the computation time. The number of memory transfers could be decreased using GPUs with larger memory, as now featured even in some notebooks.

The use of GPUs for fast ultrasound simulation is indeed promising and paves the way for the investigation of new applications. For example, the so far

prohibitively long parameter sweep that is needed for optimization purposes becomes possible. Pasovic et al. [25] have recently discussed the advantages of limiting the level of second harmonics created during nonlinear propagation. In this technique, an optimal limitation can be achieved through a specific probe excitation, provided the amount of second harmonic that must be compensated is previously quantified. A fast calculation of the second harmonic component, as is possible with the GPUs, reveals the possibility of adapting the second harmonic reduction during clinical exams. Indeed, in these cases, the probe or the medium movements decrease the resulting reduction. If quick simulations are possible, the optimization of the second harmonic reduction can be conducted concurrently with the exam in order to adapt the reduction in real time.

One known limitation of the GASM concerns the simulation bandwidth. For example, it is surely not adequate for the needs of cMUT transducers [26], a new ultrasound transducer technology that works with high-wideband signals. One possibility would be dividing the initial frequency band into multiple sub-bands and running the GASM over each of them. Of course, one should check whether the total simulation time becomes comparable to that of finite difference methods.

The GPU programming of the GASM shows a very promising opportunity in time reduction simulation in ultrasound. The GASM is the first method in ultrasound that has been tested on a GPU and the results obtained show several opportunities for future simulation tools and applications.

#### Acknowledgements

Special thanks are extended to ANR-07 TecSan-015-01 MONITHER for financial support. FV was financially supported by the Franco-Italian University with a VINCI and a Gallilée grant and by the Rhone-Alpes region with an Explora/Doc grant.

#### Author details

<sup>1</sup>Université de Lyon, CREATIS; CNRS UMR5220; INSERM U1044; INSA-Lyon; Université Lyon 1; 7 av Jean Capelle, 69621 Villeurbanne, France <sup>2</sup>Electronics and Telecommunications Department, Università degli Studi di Firenze, Via Santa Marta 3, 50139 Firenze, Italy

#### Competing interests

The authors declare that they have no competing interests.

Received: 28 February 2011 Accepted: 1 November 2011

Published: 1 November 2011

#### References

1. Averkiou MA, Roundhill DN, Powers JE: A new imaging technique based on the nonlinear properties of tissues. *IEEE Ultrasonics Symposium* 1997, **2**:1561-1566.
2. Lee Y-S, Hamilton MF: Time-domain modeling of pulsed finite-amplitude sound beams. *J Acoust Soc Am* 1995, **97**(2):906-917.
3. Christopher PT, Parker KJ: New approaches to the linear propagation of acoustic fields. *J Acoust Soc Am* 1991, **90**(1):507-521.
4. Varslot T, Taraldsen G: Computer simulation of forward wave propagation in soft tissue. *IEEE Trans Ultrason Ferroelectr Freq Control* 2005, **52**(9):1473-82.
5. Varslot T, Masoy S-E: Forward propagation of acoustic pressure pulses in 3D soft biological tissue. *Model Identification Control* 2006, **27**:181-200.
6. Dursun S, Varslot T, Johansen T, Angelsen BAJ, Torp H: Fast 3D simulation of 2nd harmonic ultrasound field from arbitrary transducer geometries. *IEEE Ultrasonics Symposium* 2005, 1964-1967.
7. Yan X, Hamilton MF: Angular spectrum decomposition analysis of second harmonic ultrasound propagation and its relation to tissue harmonic imaging workshop. *4th International Workshop on Ultrasonic and Advances Methods for Nondestructive Testing and Material Characterization* 2006.
8. Wojcik J, Kujawska T, Nowicki A, Lewin PA: Fast prediction of pulsed nonlinear acoustic fields from clinically relevant sources using time-averaged wave envelope approach: comparison of numerical simulations and experimental results. *Ultrasonics* 2008, **48**(8):707-15.
9. Du Y, Jensen H, Jensen JA: Simulation of second harmonic ultrasound fields. *IEEE Ultrasonics Symposium San Diego*; 2010.
10. Chang L-W, Hsu K-H, Li P-C: GPU-based color Doppler ultrasound processing. *IEEE International Ultrasonics Symposium* 2009, 1836-1839.
11. Kiss G, Nielsen E, Orderud F, Torp HG: Performance optimization of block matching in 3D echocardiography. *IEEE International Ultrasonics Symposium* 2009, 1403-1406.
12. Romero D, Martinez-Graullera O, Martin CJ, Higuera RT, Octavio A: Using GPUs for beamforming acceleration on SAFT imaging. *IEEE International Ultrasonics Symposium* 2009, 1334-1337.
13. Elnokrashy AF, Elmalky AA, Hosny TM, Ellah MA, Megawer A, Elsebai A, Youssef ABM, Kadah YM: GPU-based reconstruction and display for 4D ultrasound data. *IEEE International Ultrasonics Symposium* 2009, 189-192.
14. Kiss G, Steen E, Asen JP, Torp HG: GPU volume rendering in 3D echocardiography: real-time pre-processing and ray-casting. *IEEE International Ultrasonics Symposium* 2010, 193-196.
15. Kutter O, Shams R, Navab N: Visualization and GPU-accelerated simulation of medical ultrasound from CT images. *Comput Methods Prog Biomed* 2009, **94**(3):250-266.
16. Shams R, Luna F, Hartley RI: An algorithm for efficient computation of spatial impulse response on the GPU with application in ultrasound simulation. *2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro* 2011, 45-51.
17. Pinton G, Dahl J, Rosenzweig S, Trahey G: A heterogeneous nonlinear attenuating full-wave model of ultrasound. *IEEE Trans Ultrason Ferroelectr Freq Control* 2009, **56**(3):474-488.
18. Karamalis A, Wein W, Navab N: Fast ultrasound image simulation using the westervelt equation. *Medical Image Computing and Computer-Assisted Intervention Springer-Verlag, Beijing*; 2010.
19. Varray F, Ramalli A, Cachard C, Tortoli P, Basset O: Fundamental and second-harmonic ultrasound field computation of inhomogeneous nonlinear medium with a generalized angular spectrum method. *IEEE Trans. Ultrason Ferroelectr Freq Control* 2011, **58**(7):1366-1376.
20. Du G, Breazeale MA: The ultrasonic field of a Gaussian transducer. *J Acoust Soc Am* 1985, **78**(6):2083-2086.
21. Belgroune D, de Belleval JF, Djelouah H: Modelling of the ultrasonic field by the angular spectrum method in presence of interface. *Ultrasonics* 2002, **40**(1-8):297-302.
22. Frigo M, Johnson SG: The design and implementation of FFTW3. *Proc IEEE* 2005, **93**(2):216-231.
23. Jensen JA: Field: a program for simulating ultrasound systems. *10th Nordic-Baltic Conference on Biomedical Imaging Published in Medical & Biological Engineering & Computing* 1996, 351-353.
24. Pasovic M, Danilouchkine M, Van Neer P, Basset O, Cachard C, Van der Steen AFW, De Jong N: Angular spectrum method for the estimation of the lateral profile of the ultrasound pressure field in the super harmonic band. *IEEE Ultrasonics Symposium, Rome* 2009.
25. Pasovic M, Danilouchkine M, Matte G, van der Steen AFW, Basset O, de Jong N, Cachard C: Broadband reduction of the second harmonic distortion during nonlinear ultrasound wave propagation. *Ultrasound Med Biol* 2010, **36**(10):1568-1580.
26. Mills DM: Medical imaging with capacitive micromachined ultrasound transducer (cMUT) arrays. *Ultrasonics Symposium, 2004 IEEE* 2004, 1:384-390.

doi:10.1186/1687-5281-2011-17

Cite this article as: Varray et al.: Simulation of ultrasound nonlinear propagation on GPU using a generalized angular spectrum method. *EURASIP Journal on Image and Video Processing* 2011 **2011**:17.