Vu et al. Visual Computing for Industry, Biomedicine, and Art (2018) 1:2
https://doi.org/10.1186/s42492-018-0002-5

Visual Computing for Industry, Biomedicine, and Art

RESEARCH Open Access

Optimizing photoacoustic image reconstruction using cross-platform parallel computation

Tri Vu, Yuehang Wang and Jun Xia*

Abstract

Three-dimensional (3D) image reconstruction involves the computations of an extensive amount of data that leads to tremendous processing time. Therefore, optimization is crucially needed to improve the performance and efficiency. With the widespread use of graphics processing units (GPU), parallel computing is transforming this arduous reconstruction process for numerous imaging modalities, and photoacoustic computed tomography (PACT) is not an exception. Existing works have investigated GPU-based optimization on photoacoustic microscopy (PAM) and PACT reconstruction using compute unified device architecture (CUDA) on either C++ or MATLAB only. However, our study is the first that uses cross-platform GPU computation. It maintains the simplicity of MATLAB, while improves the speed through CUDA/C++ – based MATLAB converted functions called MEXCUDA. Compared to a purely MATLAB with GPU approach, our cross-platform method improves the speed five times. Because MATLAB is widely used in PAM and PACT, this study will open up new avenues for photoacoustic image reconstruction and relevant real-time imaging applications.

Keywords: Photoacoustic computed tomography, Graphics processing units, Parallel computation, Focal-line back-projection algorithm, MATLAB, Optical imaging

Background

Photoacoustic imaging is an emerging modality, which is well-known for overcoming the light diffusion limit by converging light absorption into sound [1]. Upon irradiation by laser or radiofrequency pulses, tissue will experience thermos-elastic expansion, which generates acoustic waves to be detected by transducers. Capitalizing on non-ionizing light illumination and rich optical contrasts, photoacoustic imaging possesses advantages in term of safety, penetration depth, and tissue contrast [2]. Photoacoustic computed tomography (PACT), in particular, employs higher energy pulses and wide-field scanning and is capable of capturing 3D structures in a wide range of scales, from vasculatures to organs [3, 4]. This character gives PACT an outstanding advantage over other tomography modalities [5].

Despite its immense possibility [6–8], PACT is limited by the extensive 3D computation. For example, to reconstruct a $200 \times 430 \times 200$ matrix, it takes half an hour on GPU-based MATLAB on our PC with an NVIDIA Titan X, using the focal-line-based 3D reconstruction algorithm [9]. Even with the fact that MATLAB is not time-efficient, this processing time is still considerable, placing a burden on the "pipeline" of 3D PA studies. Since reconstruction is the very "front door" component in this "pipeline", long reconstruction time leads to delay in the overall research process.

Current efforts on shortening PA reconstruction range from algorithm development to hardware improvement. In terms of algorithm development, fast Fourier transform-based (FFT) reconstruction [10] has succeeded at improving reconstruction speed. On the other hand, in terms of hardware enhancement, with the recent boom in graphics processing units (GPU), parallel computation has been widely used in various medical tomography modalities, such as PA [11–14], CT and MRI [15–18]. Because PA reconstruction involves mostly linear computation which is straightforward for being parallelized, GPU becomes a suitable solution for

* Correspondence: junxia@buffalo.edu
Department of Biomedical Engineering, University at Buffalo, The State University of New York, Buffalo, USA

Springer Open

© The Author(s). 2018 **Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

improving the computation time [11–14]. Kang et al. [11] combined both FFT reconstruction with GPU to show a significant improvement of 60 times compared to single-thread CPU on optical-resolution photoacoustic microscopy (OR-PAM) with 500 × 500 pixels. Impressive improve in performance also proved in 3D reconstruction. For instance, Wang et al. implemented GPU-based image reconstruction on C and demonstrated an improvement of 1000 times in comparison with CPU [19]. Luis et al. even managed to perform 4D PA imaging with 120 × 120 × 100 voxels and achieved a speed of 51 frames per second [20]. However, all these studies focused on the reconstruction efficiency and neglected the front-end simplicity of user interaction, which is also important in PA studies.

To improve the user-friendliness of image reconstruction, here we propose a cross-platform image reconstruction approach. Our solution is different from previous studies in a sense that it spans across two programming platforms – MATLAB and C++ on CUDA API. This MATLAB/C++/CUDA code (MCCC) combines the simplicity of MATLAB and the time-efficiency of C++. It can tremendously assist PA research because most of current PA systems heavily depend on MATLAB. In details, the reconstruction code is back-projection-based, with pre- and post-processing steps performed in MATLAB and reconstruction loops executed in CUDA/C++, through MEXCUDA functions. Validating images are then reconstructed using the MATLAB/CUDA-without-C++ code (MCC), MATLAB-without-GPU code (MWGC), and our MCCC in this study. MCC does not perform any computation in C++, and MWGC processes all the steps on CPU only. They are used to compare with MCCC to see if our cross-platform method reduces the reconstruction time. Successfully, our solution is able to shorten this processing time to one-fifth while keeping the same image quality comparing to the MCC.

Methods

Reconstruction method – focal-line-based back-projection algorithm

In PACT, the universal back-projection (UBP) is frequently used for 3D image reconstruction [21]. Details of this reconstruction method are described by the following formula:

$$p_0(\vec{r}) = \frac{1}{\Omega_0}\int_S d\Omega\left[2p(\vec{r_d},t) - 2t\frac{\partial p(\vec{r_d},t)}{\partial t}\right]\Big|_{t=\left|\vec{r_d}-\vec{r}\right|/v_s}$$

Here, $p_0(\vec{r})$ is the initial PA pressure at $\vec{r}$, $p(\vec{r_d},t)$ is the acoustic pressure at $\vec{r_d}$, and delay time t is calculated from the travel time $|\vec{r_d}-\vec{r}\,|/v_s$, in which v_s is the speed of sound in tissue (1.54 m/msec). Ω_0 is the solid angle spanning over the transducer surface S. The universal back-projection algorithm is developed based on point-like transducers and is inaccurate for focused transducers, such as linear transducer arrays with a focus along the axial direction. In this case, because of the element aperture, time delay cannot be computed directly from the point source to the center of the element. The focal-line reconstruction algorithm addresses this issue by utilizing a focal line which goes through the foci of all transducer elements. The travel path (time of arrival) of any point in 3D space is quantified based on its intersection with the focal line: only the path that goes across the focal line gives the strongest response in the transducer. Detailed descriptions of this method can be found in [9, 22].

MEXCUDA function generation

As aforementioned, MATLAB is used as the main platform for pre- and post-processing the data and all the extensive computation process is performed in C++. Such that, we need to establish a "gateway" between CUDA/C++ and MATLAB. MEXCUDA function offers a perfect solution for this connection. It is a convenient way to take input from MATLAB to C++, perform calculation in C++, and then take the output back to MATLAB. In details, MEXCUDA is the expansion of MATLAB mex function that utilizes C/C++ for execution using C++ MEX API. The difference between mex and MEXCUDA is that MEXCUDA is compiled by the NVIDIA CUDA compiler (nvcc), enabling GPU execution on C++ for improved performance.

We first need to generate a MEXCUDA function before calling it in MATLAB. The source code for the MEXCUDA function is a CU file which is written in C++ for CUDA. The CU file has the following main building blocks. The first block is initialization with two purposes. First, it prepares the code with MathWorks' GPU library by calling mxInitGPU from the mxGPU API. Secondly, it creates mxGPUArray objects (mxGPUArray is a CUDA class to contain GPU arrays) to store gpuArray inputs from MATLAB and an output matrix "*pa_img*" representing the reconstructed image. The next block of code is parallel computation. It contains several kernel functions on the device code to calculate *pa_img* from the input mxGPUArray objects in parallel. The last block of the CU code is finalization. It includes functions to deliver *pa_img* back to MATLAB code and to destroy the GPU matrices to save memory. From this source code, we create the compiled MEXCUDA function by using the mexcuda command in MATLAB. This final MEXCUDA function is in mexw64 type, which is a nvcc-compiled code for the 64-bit Windows operating system. This function can be called directly in Matlab as a subfunction.

The workflow of a function execution by MEXCUDA is demonstrated in Fig. 1. First, in the MATLAB front-end code, users load raw data, convert CPU-based matrices into GPU matrices, and set reconstruction parameters. Then, users send inputs to MEXCUDA function. After executing through the building blocks mentioned above, this function returns the output as the final reconstructed image to MATLAB. Finally, with post-processing steps in MATLAB, users are able to visualize and examine the reconstructed 3D structure.

Heterogeneous computing in CUDA/C++

The process flow executed in C++ employs a widely-known programming method called heterogeneous computing in order to maximize the performance. GPU, despite having excellent computing ability by calculating each matrix value in parallel, cannot perform both traditional serial and CPU-based tasks effectively, such as checking input compatibility, pre-allocating memory, and creating output arrays. On the other hand, CPU is faster at handling these steps so it is better suited for pre- and post-processing data. Such that, CPU is employed in the initialization and finalization blocks, while GPU is exploited in the parallel computation block. This processing flow is presented in Fig. 2.

Validating experiments

To evaluate the efficiency of the optimized code, we scanned a breast of a healthy volunteer to acquire 3D vascular data. The human imaging study was performed in compliance with the University at Buffalo IRB protocols. The PACT imaging system contains three main parts: a 10-ns-pulsed Nd:YAG laser with 10 Hz pulse repetition rate and 1064 nm output wavelength, a customized linear array with 128 elements and 2.25 MHz central frequency, and a Verasonics' Vantage data acquisition system with 128 receive channels. The light illumination was achieved through a bifurcated fiber bundle with 1.1-cm-diameter circular input and two 7.5-cm-length line outputs (Light CAM #2, Schott Fostec). During the experiment, the input laser energy was around 800 mJ/pulse and the efficiency of the fiber bundle is 60%, so that the laser output from the fiber bundle is around 480 mJ/pulse. Since the size of the laser beam on the object's surface was approximately 2.5 cm × 8.0 cm, the laser intensity is 30 mJ/cm^2, which is much lower than the safety limit of 100 mJ/cm^2 [23]. The transducer was scanned along the elevation direction over 40 mm at 0.1 mm step size. The entire imaging area is 8.6 cm (lateral width of the probe) × 4 cm (scanning distance). A schematic of the experimental setup is illustrated in Fig. 3. Following data collection, we performed 3D focal-line reconstruction with MCCC, MCC and MWGC for comparison.

Results

We reconstruct the image using MCC, MCCC and MWGC methods for comparison of processing time and image quality. All the reconstructions are carried out in our PC with an NVIDIA Titan X GPU (Pascal architecture) and Intel Core i5-6400 CPU.

In terms of reconstruction time, even though it is already supported by GPU for parallel programming, MCC reconstruction still shows a costly computing time. It takes more than 30 min with a resolution factor (RF) of five for a volume of 200 × 430 × 200 voxels. Here, RF is the reciprocal of the voxel size (in mm). Reducing this number can reduce the reconstruction time to 400 s as shown in Fig. 4 with the loss of resolution as a tradeoff.

The results clearly show that C++ has played a vital role to shorten the reconstruction time. Overall, for RF of 5, computing time in MCCC is reduced by almost five-fold in comparison with MCC, from 33 to 7 min. Expectably, both of the codes with GPU support (MCC and MCCC) outperform the reconstruction without GPU (MWGC) which takes up to 1376 min as shown

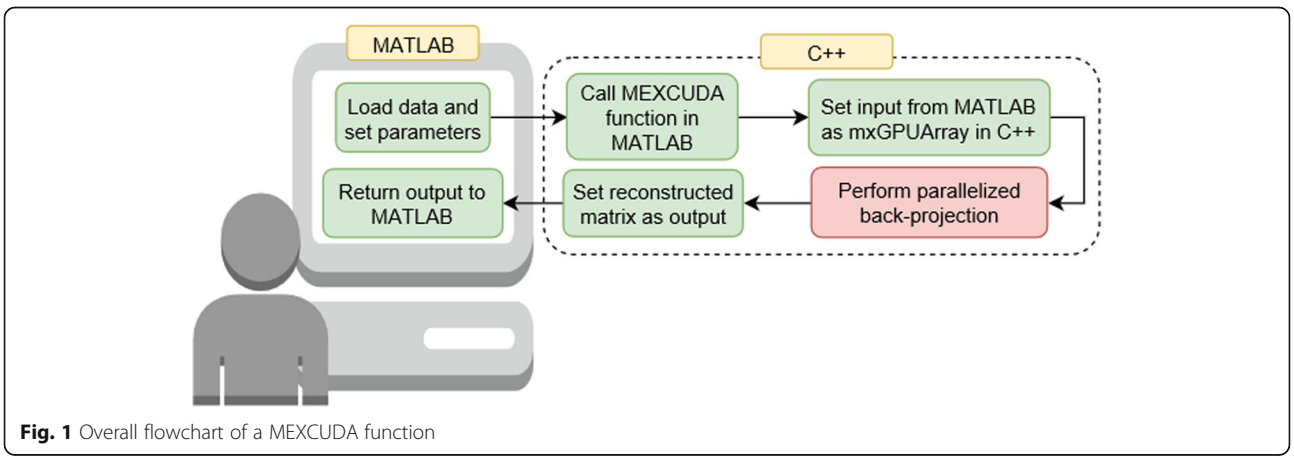

Fig. 1 Overall flowchart of a MEXCUDA function

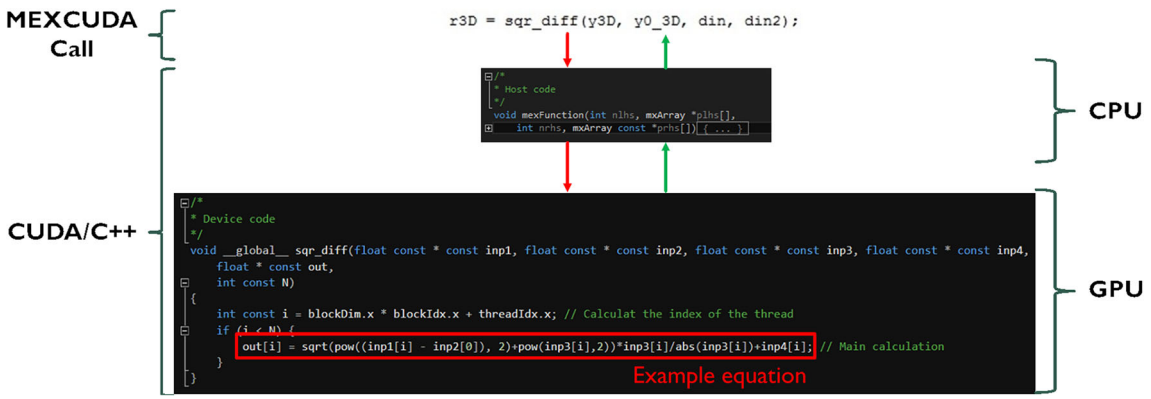

Fig. 2 Example process flow of the heterogeneous computing. In this figure, we create a sample MEXCUDA function for calculating the radius matrix of a scanned region to each transducer elements. The host code (CPU) is in charge of initialization and finalization blocks such as reading the input from MATLAB, performing condition checks, and allocating memory for input and output data. Device code in GPU is responsible for computing the radius matrix in parallel from required inputs (parallel computation block)

in Fig. 4(a). At RF of 2, MCCC has ten times shorter processing time than the duration of MCC as shown in Fig. 4(b). In terms of image quality, because the reconstruction methods are the same, there are no changes from those images created by MCC and MCCC as indicated by Fig. 5. This fact proves that there is no tradeoff between reconstruction accuracy and processing time. From the perspective of the users, as aforementioned, there is no need for modification of parameters or calculations in the source code in C++ because it is used as a predefined sub-function. With this fact, the simplicity nature of the front-end MATLAB code is maintained.

Discussion and conclusion

To summarize, in this paper, we propose a novel way to optimize 3D reconstruction for PACT using cross-platform MATLAB/C++ code on CUDA. Our approach, utilizing C++/GPU reconstruction function, manages to significantly reduce the reconstruction time by five times compared with the performance of the MATLAB/GPU code. On the other hand, it maintains the simplicity of user interaction in MATLAB front-end side. Our method paves the way for future 3D reconstruction optimization for PACT on cross-platform MATLAB/C++ that benefits further PACT research which depends heavily on MATLAB.

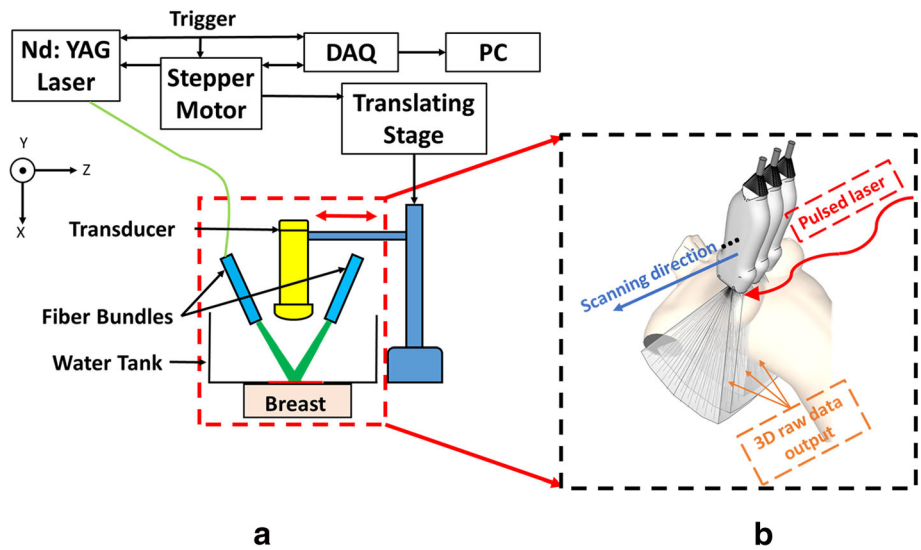

Fig. 3 Schematic drawing of PACT setup. **a** A 2D schematic showing all major components. **b** A 3D illustration of the scanning process. The laser beam causes the tissue to expand and release acoustic waves which are captured by the transducer. This process happens continuously along the moving (scanning) direction

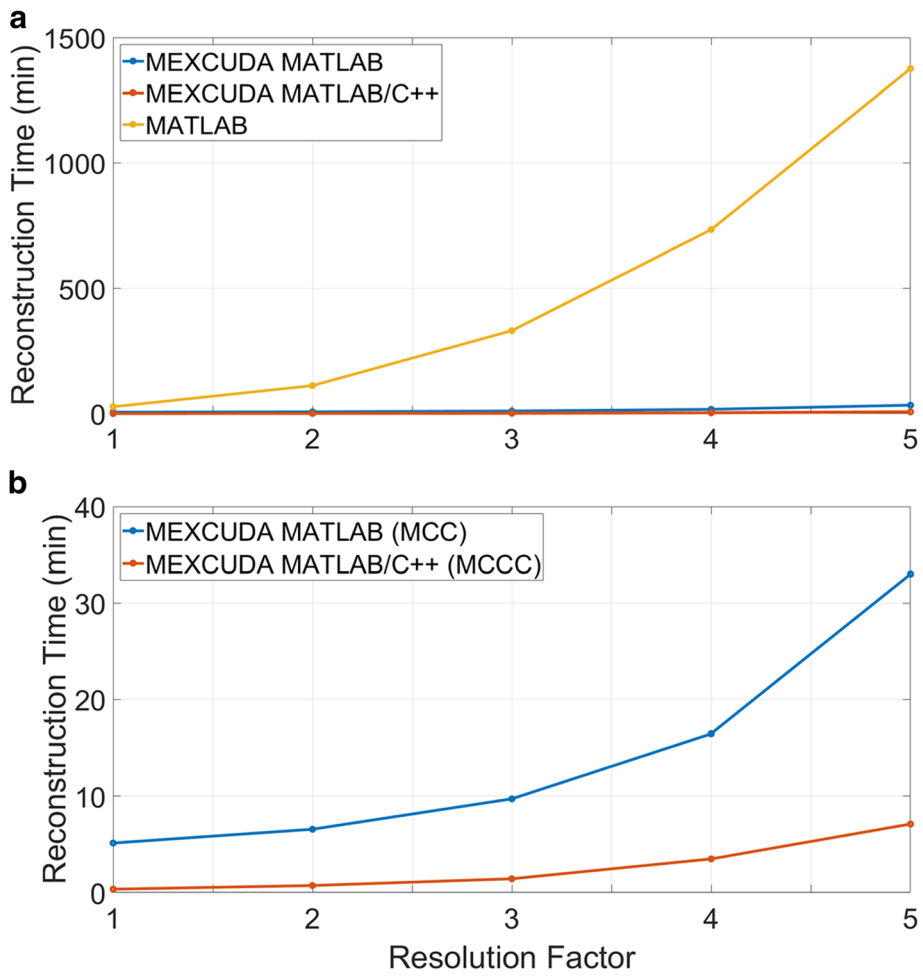

Fig. 4 **a** Comparison of reconstruction time between MCC, MCCC and MWGC and **b** a close look into reconstruction time difference between MCC and MCCC codes with different RF

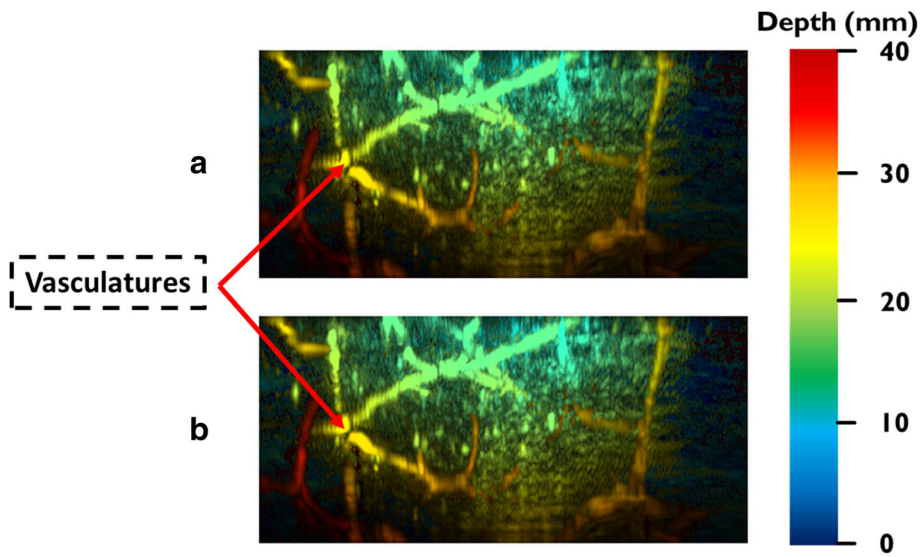

Fig. 5 Comparison of depth-encoded photoacoustic images reconstructed by the (**a**) MCC and (**b**) MCCC

Future work for this project will focus on further decreasing the reconstruction time by cutting down the number of iterations in the source code. The current reconstruction is still processed through a significant amount of loops based on the number of transducer elements and scanning lines. Instead of going through 128 (number of elements) × 400 (number of lines) loops, we should find a solution to perform calculation all at once if possible. For example, all the input data for each loop can be allocated to all available GPU memory and be executed in parallel. However, the limitation of this approach is that a huge amount of memory will need to be deployed, making it only viable for small 3D PA structures. Other than that, reducing the number of iterations can be achieved by having only either 128 loops based on transducer elements or 400 loops based on lines.

Acknowledgements
This study is supported in part by the Career Catalyst Research Grant from the Susan G. Komen Foundation and the Clinical and Translational Science Pilot Study Award from the National Institutes of Health. The Titan X Pascal used for this research was donated by the NVIDIA Corporation.

Authors' contributions
TV and JX conceived the study. TV wrote the Matlab and C++ codes and YW conducted the experimental study. JX guided the overall direction of the study. All authors read and approved the final manuscript.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Received: 1 April 2018 Accepted: 16 June 2018
Published online: 05 September 2018

References

1. Beard P. Biomedical photoacoustic imaging. Interface focus. 2011;1(4):602–31.
2. Zhang E, Laufer J, Pedley R, Beard P. In vivo high-resolution 3D photoacoustic imaging of superficial vascular anatomy. Phys Med Biol. 2009;54(4):1035.
3. Jeon M, Kim J, Kim C. Multiplane spectroscopic whole-body photoacoustic imaging of small animals in vivo. Med Biol Eng Comput. 2016;54(2–3):283–94.
4. Wang LV, Yao J. A practical guide to photoacoustic tomography in the life sciences. Nat Methods. 2016;13(8):627.
5. Jathoul AP, Laufer J, Ogunlade O, Treeby B, Cox B, Zhang E, et al. Deep in vivo photoacoustic imaging of mammalian tissues using a tyrosinase-based genetic reporter. Nat Photonics. 2015;9(4):239.
6. Li ML, Oh JT, Xie X, Ku G, Wang W, Li C, et al. Simultaneous molecular and hypoxia imaging of brain tumors in vivo using spectroscopic photoacoustic tomography. Proc IEEE. 2008;96(3):481–9.
7. Shao Q, Morgounova E, Jiang C, Choi J-H, Bischof JC, Ashkenazi S. In vivo photoacoustic lifetime imaging of tumor hypoxia in small animals. J Biomed Opt. 2013;18(7):076019.
8. Xia J, Chatni MR, Maslov KI, Guo Z, Wang K, Anastasio MA, et al. Whole-body ring-shaped confocal photoacoustic computed tomography of small animals in vivo. J Biomed Opt. 2012;17:050506.
9. Xia J, Guo Z, Maslov K, Aguirre A, Zhu Q, Percival C, et al. Three-dimensional photoacoustic tomography based on the focal-line concept. J Biomed Opt. 2011;16(9):090505.
10. Köstli KP, Beard PC. Two-dimensional photoacoustic imaging by use of Fourier-transform image reconstruction and a detector with an anisotropic response. Appl Opt. 2003;42(10):1899–908.
11. Kang H, Lee SW, Lee E, Kim SH, Lee TG. Real-time GPU-accelerated processing and volumetric display for wide-field laser-scanning optical-resolution photoacoustic microscopy. Biomed Opt Express. 2015;6(12):4650–60.
12. Kruger RA, Kuzmiak CM, Lam RB, Reinecke DR, Del Rio SP, Steed D. Dedicated 3D photoacoustic breast imaging. Med Phys. 2013;40(11)113301:1–8.
13. Treeby BE, Cox BT. k-Wave: MATLAB toolbox for the simulation and reconstruction of photoacoustic wave fields. J Biomed Opt. 2010;15(2):021314.
14. Yuan J, Xu G, Yu Y, Zhou Y, Carson PL, Wang X, et al. Real-time photoacoustic and ultrasound dual-modality imaging system facilitated with graphics processing unit and code parallel optimization. J Biomed Opt. 2013;18(8):086001.
15. Jia X, Lou Y, Li R, Song WY, Jiang SB. GPU-based fast cone beam CT reconstruction from undersampled and noisy projection data via total variation. Med Phys. 2010;37(4):1757–60.
16. Scherl H, Keck B, Kowarschik M, Hornegger J, editors. Fast GPU-based CT reconstruction using the common unified device architecture (CUDA). Nuclear Science Symposium Conference Record, 2007 NSS'07 IEEE. Honolulu: IEEE; 2007.
17. Smith DS, Gore JC, Yankeelov TE, Welch EB. Real-time compressive sensing MRI reconstruction using GPU computing and split Bregman methods. Int J Biomed Imaging. 2012;2012(864827):1–6.
18. Stone SS, Haldar JP, Tsao SC, Sutton B, Liang Z-P. Accelerating advanced MRI reconstructions on GPUs. J Parallel Distrib Comput. 2008;68(10):1307–18.
19. Wang K, Huang C, Kao YJ, Chou CY, Oraevsky AA, Anastasio MA. Accelerating image reconstruction in three-dimensional optoacoustic tomography on graphics processing units. Med Phys. 2013;40(023301):1–15.
20. Dean-Ben XL, Ozbek A, Razansky D. Volumetric real-time tracking of peripheral human vasculature with GPU-accelerated three-dimensional optoacoustic tomography. IEEE Trans Med Imaging. 2013;32(11):2050–5.
21. Xu M, Wang LV. Universal back-projection algorithm for photoacoustic computed tomography. Phys Rev E. 2005;71(1):016706.
22. Wang D, Wang Y, Zhou Y, Lovell JF, Xia J. Coherent-weighted three-dimensional image reconstruction in linear-array-based photoacoustic tomography. Biomed Opt Express. 2016;7(5):1957–65.
23. Standard A. American national standard for the safe use of lasers. Z136. 2000;1:2007–1.

Submit your manuscript to a SpringerOpen journal and benefit from:

► Convenient online submission
► Rigorous peer review
► Open access: articles freely available online
► High visibility within the field
► Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com