INTERACTIVE HIGH RESOLUTION RECONSTRUCTION OF 3D ULTRASOUND VOLUMES ON THE GPU

Marwan Abdellah†, *Asem Abdelaziz and Ayman Eldeib*

Biomedical Engineering Department, Cairo University, Egypt

ABSTRACT

Since its inception, ultrasound has been recognized to be one of the most significant technologies in clinical imaging. The rapid and continuous development of ultrasound scanning technologies has added another challenge of rendering relatively large scale 3D volumes with high resolution. Moreover, some applications have required particular high quality ultrasound rendering at interactive frame rate for improved interpretation of the data. Based on the vast computing power of the modern Graphics Processing Units (GPUs), we present an efficient ultrasound reconstruction pipeline that is capable of rendering high quality and high resolution ultrasound images at interactive frame rates for large scale volumes. The reconstruction results of our rendering pipeline have been applied on a real fetal ultrasound data. The performance benchmarks of our pipeline demonstrate its capability of rendering high quality ultrasound images for 1024^3 volumes with resolutions of 2048^2 at interactive frame rates using a recent commodity GPU.

***Index Terms*—** High quality 3D ultrasound, interactive volume reconstruction, off-screen scan conversion, CUDA.

1. INTRODUCTION & BACKGROUND

The current trends in diagnostic and clinical medicine use ultrasound imaging frequently in comparison to other imaging modalities [1, 2]. This fact is well interpreted in terms of its high spatial resolution, non-invasiveness, high availability, real-time interactivity, and overall favorable costs [3]. As a consequence, the impact of ultrasound has been reflected in a broad spectrum of clinical applications including the imaging of heart, abdomen, pelvis in addition to its significant usage in several obstetric and gynecological examinations [4, 5]. Ultrasound has progressively become a standard diagnostic tool for every pregnant woman who is interested to continuously monitor the state of her fetus during pregnancy.

Since the inception of this imaging technology, interactive ultrasound volume visualization has been recognized as an extremely interesting and competing topic for the computer graphics community. The research pursued in this direction has focused on preserving the interactivity of the rendering on one hand, and improving the quality of the reconstructed images on the other hand. The current state-of-the-art ultrasound devices are capable of delivering high resolution fetal images due to their advanced scanning probes that could generate large scale volumetric data in real-time [6]. Nevertheless, the present visualization pipelines are known to be limited in that regard either in their rendering performance for displaying high resolution volumes or in the reconstruction quality of high resolution images. In contrary to other imaging modalities such as CT, SPECT or MRI, this limitation is only relevant to ultrasound due to the way the volume acquisition procedure is performed. This acquisition results in arranging the volume in what is called a *pyramidal grid*, which requires further computationally-intensive processing to render an undistorted image in comparison to the conventional Cartesian grids that are used to sample the volumes obtained from CT scanners for instance [7]. Rendering high quality ultrasound images at interactive frame rates was always demanded by the physicians for effective diagnosis. During the last three decades, several algorithms and improvements have been developed to meet this demand. More recently, ultrasound visualization pipelines have been designed and implemented with several rendering algorithms on the GPUs to attain more interactivity by exploiting their parallel architecture [6]. However, these implementations were not scalable enough to preserve the interactivity for rendering large scale volumetric data that are acquired with recent ultrasound probes [8]. In turn, the generation of high quality ultrasound images at that scale requires designing new optimized GPU-based parallel execution engines that would be capable of handling high resolution volume data interactively. This paper features an accelerated GPU-based ultrasound visualization pipeline that can render relatively large scale ultrasound data (1024^3 volumes) interactively at high resolutions (up to 2048^2 images) relying on a recent commodity GPU that uses shading to scan convert the data and CUDA (Compute Unified Device Architecture) to render the volumes in real-time.

2. ULTRASOUND RENDERING PIPELINE

The proposed rendering pipeline consists of two principal phases: *volume scan conversion* and the *rendering loop*. The scan conversion process is executed in a pre-processing stage

† Corresponding author: *marwan.m.abdellah@ieee.org*

to convert the ultrasound pyramidal grid into a default uniformly sampled Cartesian grid that is required to generate non-distorted ultrasound images relying on the *ray-tracing* rendering algorithm. The rendering loop is executed on a per-frame-basis as a result of either manipulating any of the reconstruction parameters (such as transfer function, data density scaling factor) or updating the camera events (such as changing the viewing frustum or the orientation of the volume).

2.1. Volume Scan Conversion

Due to the nature of the ultrasound scanning mechanism, the volume acquisition process takes place in the polar coordinates. After the acquisition of the volume, the data is stored in a 3D truncated pyramidal grid. This step requires resampling the ultrasound data along the range direction. Consequently, the quality of the acquired data depends on the type of the scanning probe. The 4D probes are highly efficient in that regard, but they might not be affordable in certain cases. The freehand scanning systems are less expensive, however they require calibration and experienced user to reduce the resampling artifacts. In contrast, *direct volume rendering* algorithms (such as ray-casting, back-to-front blending) can be implemented only in the Cartesian space, and thus, the data must be converted from pyramidal into Cartesian grids. As seen in Figure 1, the scan conversion process is crucial for this coordinate transformation process. Sumanaweera and others have used *projective texture mapping* to accomplish this step in real-time using the Cg shading language [9, 10, 7, 11, 12]. However, this algorithm is limited in two aspects. On one hand, the texture mapping and alpha-blending reconstruction algorithm cannot produce high quality renderings compared to ray tracing. Moreover, it is only capable of handling relatively small volumes (for instance $256 \times 256 \times 32$ as shown in the results of [7]) to maintain the interactivity of the rendering process. Increasing the size of the input volume would substantially reduce the performance of the rendering pipeline.

Our rendering pipeline has extended this method to generate scan-converted volumes that could be directly used for interactive rendering of high quality and high resolution ultrasound images. This extension renders a series of 2D slices of the converted volume off-screen to a Framebuffer Object (FBO) [13]. The off-screen context is set with an orthographic frustum that has the same dimensions of the output volume. The scan-converted volume is sampled by a single moving cut-plane and the intersection of the plane with the 3D texture that contains the volume data is projected into the 2D texture that is attached to the FBO. After scanning the entire 3D texture with the clipping plane, all the generated projections are arranged in a 3D volume and uploaded to the GPU memory on the CUDA address space for shading and on-screen rendering.

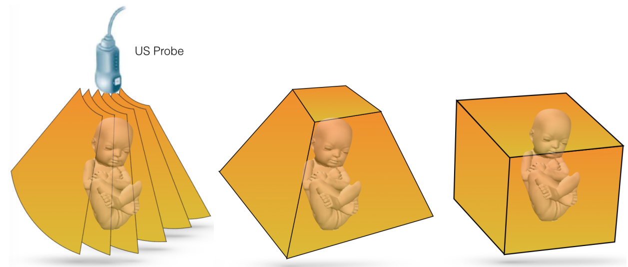

Fig. 1. The scan conversion process: the ultrasound volume is acquired in polar coordinates (left), stored in a pyramidal grid (middle), and finally converted into a Cartesian grid for the rendering stage (right).

2.2. CUDA-based Rendering

The flow of the rendering stage is adopted from an efficient pipeline that is used to generate digitally reconstructed radiographs [14, 15, 16]. The rendering loop is composed of two contexts: an *OpenGL context* and a *CUDA compute context*. Although the rendering operation is executed in the compute context relying on a CUDA compute kernel that implements the ray marching algorithm, the OpenGL context is created for a couple of reasons: (1) handling the transformations, building the model-view matrix and sending it to the constant memory of the CUDA address space every frame and (2) displaying the reconstructed ultrasound image after its rendering on an OpenGL widget considering the interoperability between OpenGL and CUDA. The CUDA compute context handles all the kernels associated with the pre- and post-processing operations in addition to the kernel that renders the final image. Figure 2 depicts the idea of rendering a scan-converted ultrasound volume in a Cartesian grid using the ray marching algorithm and Figure 3 illustrates the different components of the rendering workflow and the communication mechanisms between its different contexts.

3. RESULTS

Figure 4 shows the rendering results for the same fetal dataset using four different transfer functions. The dataset is downloaded from [7]. The performance of our rendering pipeline was measured and analyzed on a relatively high-end workstation that is shipped with an Intel Core i7 CPU (3.2 GHz), 20 GBytes of DRR3 memory and a recent commodity GPU – an NVIDIA GeForce GTX 970. This GPU contains 1560 CUDA cores (1050 MHz) and four GBytes of DDR5 memory.

Figure 5 shows the performance benchmarks of rendering a scan-converted volume with three different sizes (256^3, 512^3, and 1024^3 voxels). The sampling step of the volumes was set to 0.01 units within a unit cube that represents the bounding box of the texture data. To investigate the strong scaling performance of the rendering pipeline, the images were rendered at three different resolutions (512^2, 1024^2, and

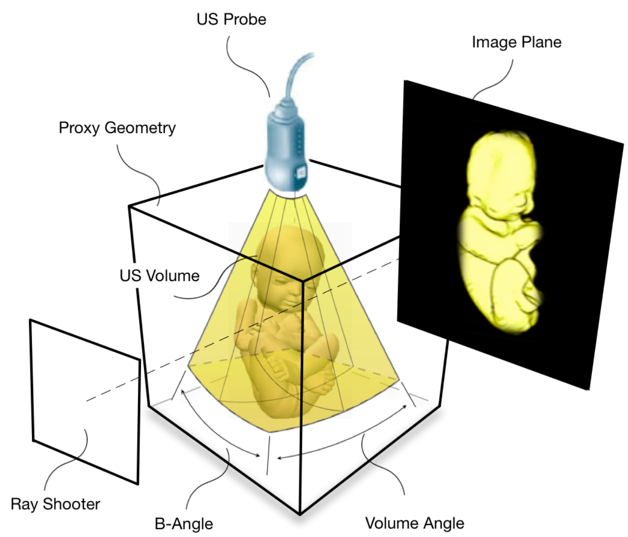

Fig. 2. Rendering a scan converted ultrasound volume with ray marching.

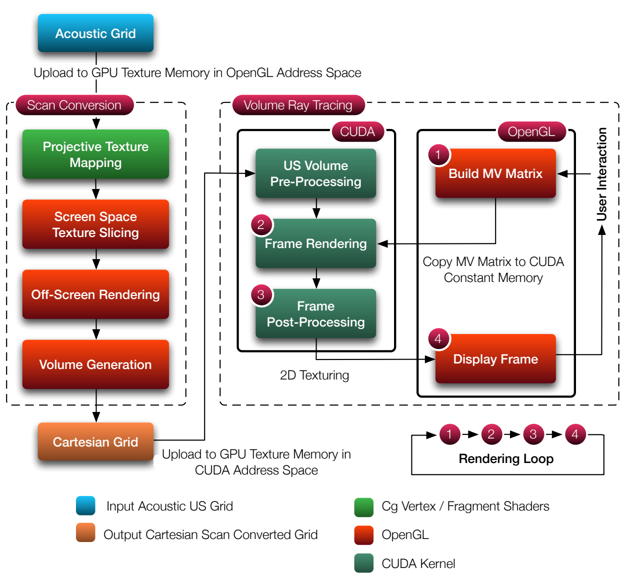

Fig. 3. A high level overview of our proposed ultrasound rendering pipeline.

2048^2 pixels). The optimization of the rendering performance was also addressed by analyzing the benchmarks at various CUDA block configurations (2×2, 4×4, 8×8, 16×16, and 32×32 threads per block). The benchmarks were recorded relying on the high precision profiling utility *cudaEventRecord* that is capable of resolving the timing differences between the various block configurations. The benchmarks in Figure 5 show that our rendering pipeline can render a 2048^2 image for a 1024^3 dataset at almost 40 frames per second. On average, the most optimized block configuration is found to 32×32 threads per block.

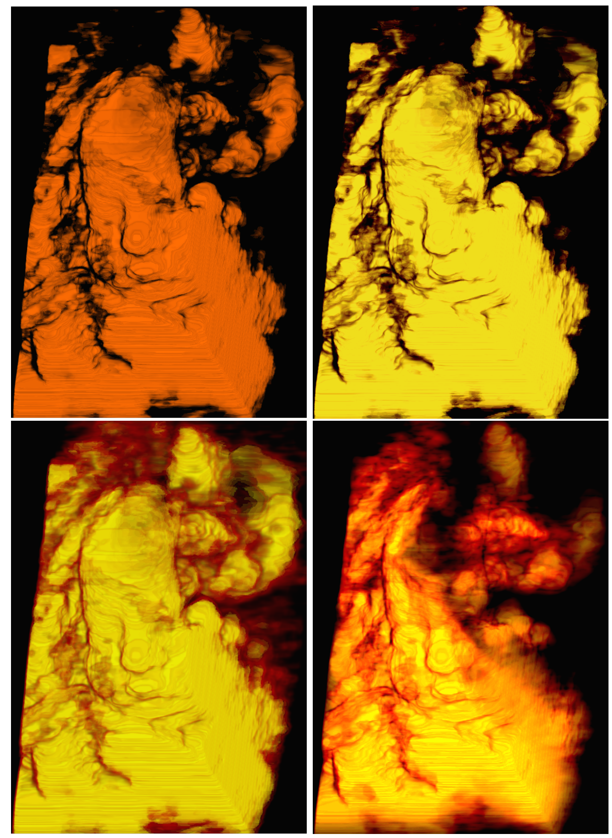

Fig. 4. Reconstruction and rendering results of a fetal ultrasound dataset using four different global transfer functions.

4. CONCLUSION & FUTURE WORK

This paper presented a high performance GPU-based rendering engine to interactively reconstruct large scale ultrasound data at high resolutions using the high quality ray tracing algorithms. The scan conversion process is implemented in an off-screen rendering context to generate high resolution volumes in a Cartesian grid. The rendering loop is implemented relying on the inter-operability mechanisms between CUDA and OpenGL. The reconstruction results are demonstrated with a realistic fetal dataset. The performance of the rendering pipeline was investigated for various CUDA block configurations. The recorded benchmarks reflect the capability of the pipeline to render a 2048^2 image for a 1024^3 dataset at 25 frames per second. The pipeline will be extended to include the support of applying different pre- and post-processing filters to maximally reduce the data acquisition and reconstruction artifacts.

5. REFERENCES

[1] Frank Gress, Thomas Savides, Brenna C Bounds, and John C Deutsch, *Atlas of Endoscopic Ultrasonography*, John Wiley & Sons, 2011.

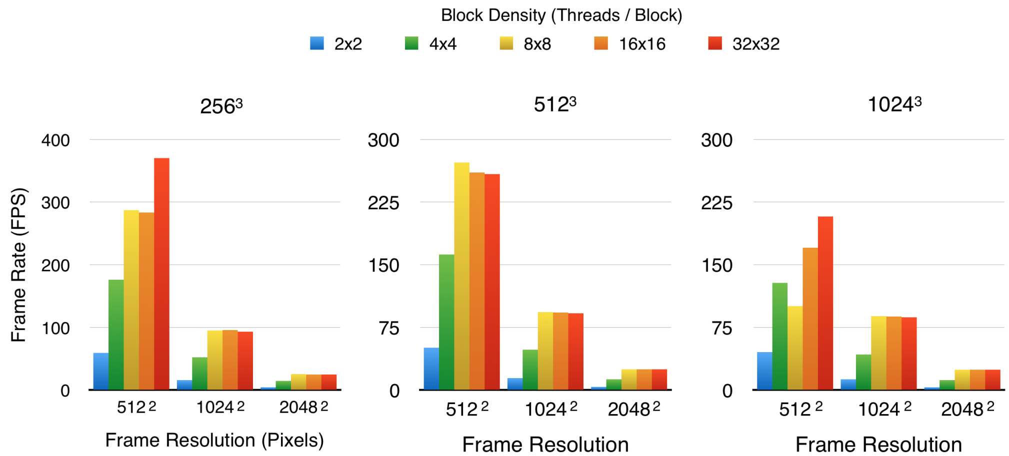

Fig. 5. Performance benchmarks for our rendering pipeline obtained for different volume dimensions (256^3, 512^3, 1024^3), frame resolutions (512^2, 1024^2, 2048^2), and CUDA block configurations (2×2, 4×4, 8×8, 16×16, 32×32).

[2] Svein Odegaard, Odd Helge Gilja, and Hans Gregersen, *Basic and new aspects of gastrointestinal ultrasonography*, vol. 3, World Scientific, 2005.

[3] Åsmund Birkeland, Veronika Solteszova, Dieter Hönigmann, Odd Helge Gilja, Svein Brekke, Timo Ropinski, and Ivan Viola, "The ultrasound visualization pipeline-a survey," *arXiv preprint arXiv:1206.3975*, 2012.

[4] Roee S Lazebnik and Terry S Desser, "Clinical 3d ultrasound imaging: beyond obstetrical applications," *Diagnostic Imaging*, vol. 1, pp. 1–6, 2007.

[5] Thomas R Nelson and Dolores H Pretorius, "Three-dimensional ultrasound imaging," *Ultrasound in medicine & biology*, vol. 24, no. 9, pp. 1243–1270, 1998.

[6] Qi Zhang, Roy Eagleson, and Terry M Peters, "Volume visualization: a technical overview with a focus on medical applications," *Journal of digital imaging*, vol. 24, no. 4, pp. 640–664, 2011.

[7] Thilaka Sumanaweera, "Applying real-time shading to 3d ultrasound visualization," *GPU Gems*, pp. 693–707, 2004.

[8] Anders Eklund, Paul Dufort, Daniel Forsberg, and Stephen M LaConte, "Medical image processing on the GPU–past, present and future," *Medical image analysis*, vol. 17, no. 8, pp. 1073–1094, 2013.

[9] Cass Everitt, "Projective texture mapping," *White paper, NVidia Corporation*, vol. 4, 2001.

[10] William R Mark, R Steven Glanville, Kurt Akeley, and Mark J Kilgard, "Cg: A system for programming graphics hardware in a c-like language," in *ACM Transactions on Graphics (TOG)*. ACM, 2003, vol. 22, pp. 896–907.

[11] Ahmed F Elnokrashy, Ahmed Elmalky, Tamer M Hosny, Marwan Abd Ellah, Alaa Megawer, Abobakr Elsebai, Abou-Bakr M Youssef, Yasser M Kadah, et al., "GPU-based reconstruction and display for 4D ultrasound data," in *Ultrasonics Symposium (IUS), 2009 IEEE International*. IEEE, 2009, pp. 189–192.

[12] Marwan Abdellah, Alaa Megawer, and Yasser M Kadah, "Software development for low cost, high quality, real-time 4D ultrasound imaging system on personal computers (PCs)," in *Radio Science Conference, 2009. NRSC 2009. National*. IEEE, 2009, pp. 1–2.

[13] Simon Green, "The OpenGL framebuffer object extension," in *Game Developers Conference*, 2005, vol. 2005.

[14] Marwan Abdellah, Ayman Eldeib, and Amr Sharawi, "High performance GPU-based Fourier volume rendering," *International Journal of Biomedical Imaging*, vol. 2015, no. Article ID 590727, pp. 13–pages, 2015.

[15] Marwan Abdellah, Ayman Eldeib, and Mohamed I Owis, "GPU acceleration for digitally reconstructed radiographs using bindless texture objects and CUDA/OpenGL interoperability," in *Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE*. IEEE, 2015, pp. 4242–4245.

[16] Marwan Abdellah, Ayman Eldeib, and Mohamed I Owis, "Accelerating DRR generation using Fourier slice theorem on the GPU," in *Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE*. IEEE, 2015, pp. 4238–4241.