10.1109/ULTSYM.2015.0146

Implementation of Real-time Duplex Synthetic Aperture Ultrasonography

Martin Christian Hemmsen[1], Lee Lassen[2], Thomas Kjeldsen[2], Jesper Mosegaard[2] and Jørgen Arendt Jensen[1]

[1]Center for Fast Ultrasound Imaging, Department of Electrical Engineering,
Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark

[2]Computer Graphics Lab
Alexandra Institute, DK-8200 Aarhus N, Denmark

Abstract— **This paper presents a real-time duplex synthetic aperture imaging system, implemented on a commercially available tablet. This includes real-time wireless reception of ultrasound signals and GPU processing for B-mode and Color Flow Imaging (CFM). The objective of the work is to investigate the implementation complexity and processing demands. The image processing is performed using the principle of Synthetic Aperture Sequential Beamforming (SASB) and the flow estimator is implemented using the cross-correlation estimator. Results are evaluated using a HTC Nexus 9 tablet and a BK Medical BK3000 ultrasound scanner emulating a wireless probe. The duplex imaging setup consists of interleaved B-mode and CFM frames. The required data throughput for real-time imaging is 36.1 MB/s. The measured data throughput peaked at 39.562 MB/s, covering the requirement for real-time data transfer and overhead in the TCP/IP protocol. Benchmarking of real-time imaging showed a total processing time of 25.7 ms (39 frames/s) which is less than the acquisition time (29.4 ms). In conclusion, the proposed implementation demonstrates that both B-mode and CFM can be executed in-time for real-time ultrasound imaging and that the required bandwidth between the probe and processing unit is within the current Wi-Fi standards.**

I. Introduction

Ultrasound technology has become progressively more portable, and now includes hand-held devices designed to complement clinical examination [1], [2]. While there have been several earlier products, they all suffered from incomplete feature sets and/or compromised imaging performance. The rapid increase in processing power has allowed the development of hand-held systems based on synthetic aperture, with imaging performance equivalent to cart-based systems [3].

Synthetic aperture (SA) ultrasound imaging has several benefits compared to conventional imaging. From a SA dataset it is possible to obtain both dynamic focus in transmit and receive. This enhances the B-mode image in terms of improved contrast and resolution [4]. SA can also be used for flow imaging [5]. However, a full SA method needs to acquire individual element data, which requires huge amounts of memory, processing power, and data throughput. Synthetic Aperture Sequential Beamforming (SASB) has been suggested, for both B-mode and flow imaging, to overcome the requirement to store element data [6], [7]. Compared to a full SA setup only a single RF-line is beamformed and stored for each emission. This reduces the number of calculations and complexity significantly, and enables data transmission from probe to processing system over Wi-Fi.

In this paper we extend the implementation in [8] to include real-time duplex imaging. The objective of this work is to investigate the implementation complexity and processing demands for duplex imaging using SASB. The hypothesis is that current consumer level tablets are fast enough and capable of receiving data and execute SASB duplex imaging in real-time. The implementation is evaluated in terms of processing- and communication performance using a HTC Nexus 9 and a BK Medical BK3000 scanner emulating a wireless probe. Duplex images of a flow rig system with a gear pump generating a parabolic laminar flow profile inside a tube are presented. Additionally this setup is used for quantitative validation of the velocity estimates.

II. Materials and Methods

The basic idea of an imaging system implemented using SASB beamforming is to create a system with a data throughput small enough to allow wireless transmission in real-time. The fundamental principle of SASB is to use two separate beamformers. The first beamformer creates fixed focused scan lines and the final image is created in the second beamformer by refocusing these scan lines. Both B-mode imaging and color flow imaging can be implemented using SASB [6], [7].

A. System setup

In this work a wireless probe is emulated using a BK Medical BK3000 ultrasound scanner and a BK Medical 10L2W linear array transducer. The scanner is configured to beamform the received echo signals using a fixed receive profile and the beamformed signals are subsequently transformed to baseband IQ data. The data is then transferred using Wi-Fi to a HTC Nexus 9 for refocusing of the fixed focused scan lines and subsequent image processing for B-mode and CFM imaging.

To incorporate synthetic aperture CFM imaging the shot sequence from [8] was modified to include eight emissions transmitted sequentially and repeated 16 times after each frame of B-mode emissions. The eight CFM emissions are transmitted in different directions and used to build one high resolution synthetic aperture data frame. Consecutive data frames are correlated to estimate the flow. The B-mode frames

978-1-4799-8182-3/15/$31.00 ©2015 IEEE

TABLE I

IMAGING PARAMETERS

B-mode	
1st stage focus depth	20 mm
1st stage aperture size	42 elements
1st stage rx-apodization	Hamming
1st stage tx-apodization	Box-car
2nd stage F#	2
2nd stage apodization	Tukey, 0.5
Excitation waveform	5.7 MHz
No. emissions	183
Field of view (W × D)	45.8 mm × 60 mm
CFM	
1st stage focus depth	10 mm
1st stage aperture size	21 elements
1st stage rx-apodization	Hamming
1st stage tx-apodization	Box-car
2nd stage F#	2
2nd stage apodization	Tukey, 0.5
Excitation waveform	3.72 MHz
No. emissions	8 directions × 16 blocks
Field of view (W × D)	13.4 mm × 30 mm

consist of 183 scan lines × 1172 complex samples (0.86 MB) and the CFM frames consist of 128 emissions × 400 complex samples (0.2 MB). Imaging parameters are shown in Table I.

B. Imaging algorithms

B-mode and CFM imaging is processed interleaved. The B-mode image is generated by IQ demodulation, beamforming, amplitude detection, compression, and scan-conversion, as shown in the left column of Fig. 1 and described in detail in [9]. Similarly, the complete process of converting the IQ data into the CFM-image is shown in the right column of Fig. 1.

The CFM image is created using the same workflow of IQ demodulation and beamforming as for the B-mode image. In the present work, beamformation of the n'th CFM emission block is computed in a Cartesian grid with N_z depth samples and N_x image lines. This leads to a high resolution frame with discrete samples $H^{(n)}(x_i, z_j)$ in the region of interest. After having created N_{rep} beamformed images, stationary echoes are filtered out using the ensemble mean. An ensemble of N_{rep} SA images is then employed for velocity estimation using the method suggested in [5]. In the presence of a displacement between images separated N frames, the spatial shift can be expressed as

$$H^{(n)}(x_i, z_j) = H^{(n-N)}(x_i, z_j - \nu_z T_{prf} N), \qquad (1)$$

where ν_z is the velocity along the axial direction and T_{prf} is the repetition interval between consecutive images. The spatial shift can be estimated using cross-correlation.

For each image line at lateral position x_i, the vertical velocity component is then computed at depth z_j as

$$v_z^{(n)}(x_i, z_j) = \frac{c}{2} \frac{1}{T_{prf} N F_s} \arg\max_{\zeta} (H_{i,j}^{(n)} \star H_{i,j}^{(n+N)})(\zeta), \quad (2)$$

where F_s is the sampling frequency and c is the speed of sound. The cross-correlation in the z direction is

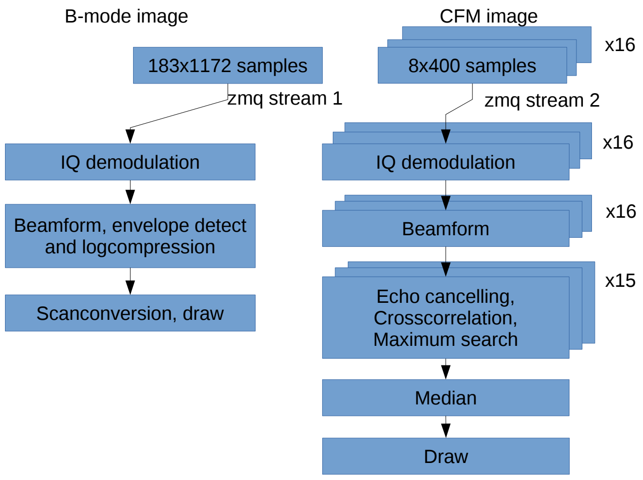

Fig. 1. System workflow. The two seperate zero message queue data streams are fed into the GPU, which in turn first processes the B-mode image and then the CFM image.

$$(H_{i,j}^{(n)} \star H_{i,j}^{(n+N)})(\zeta) = \sum_{k=j-\frac{N_W}{2}}^{j+\frac{N_W}{2}-1} W(z_k - z_j) H_{\text{EC}}^{(n)}(x_i, z_k) \times W(z_k + \zeta - z_j) H_{\text{EC}}^{(n+N)}(x_i, z_k + \zeta). \quad (3)$$

Here $W(z_k - z_j)$ is a window function that only selects N_W input samples in a range gate around the output sample z_j, and subscripts EC denote that echo cancelling filters have been applied. Finally, the velocity at (x_i, z_j) is estimated as the median of the $N_{\text{rep}} - 1$ individual estimates.

C. GPU implementation

For realtime performance to be achieved, the GPU is used to parallelize all workflow steps in Fig. 1. The GPU is programmed using the OpenGL ES 3.1 API with the individual tasks implemented as compute- or fragment shaders. Efficient GPU implementation of B-mode imaging, was described in [9], [8]. Hence, the present section mainly focuses on the remaining CFM steps.

The most complex and time-consuming part is the cross-correlation of the signals between shots. In order to optimize this, one can take advantage of compute shaders that were introduced recently with OpenGL ES 3.1.

Direct implementation of (3) requires $4(N_{\text{rep}} - 1)N_W$ memory lookups of the beamformed images H for each output pixel. This would be the only possible method for prior versions of OpenGL ES. Unfortunately, this amount of memory reads turns out to become a bottleneck. In a compute shader, however, it is possible to reduce the number of relatively slow memory reads to $N_{\text{rep}} N_W$ per output pixel.

The reduction in memory reads is achieved by arranging compute shaders in workgroups, which consist of threads that operate on collections of data as independent units, and are able to share a small amount of data between them. This shared memory can be accessed within the workgroup for reads and

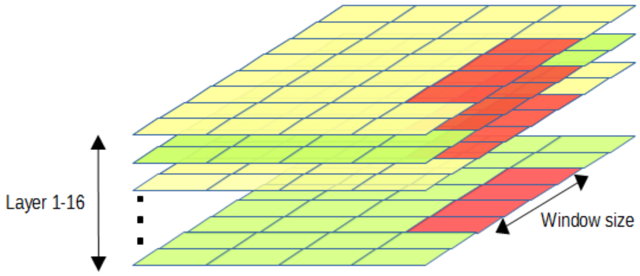

Fig. 2. The 16 2-D layers. A workgroup reads N_W samples from all the layers beamformed images into the shared memory (shown in red). The samples in the shared memory are cross-correlated for each pair of neighbouring layers.

writes much faster than regular off-chip GPU DRAM. The input data to the cross-correlation are the N_{rep} beamformed images which are stored in an array of 2D-textures as shown in Fig. 2. Given the window size N_W, 2D-workgroups are then defined to match the size of $N_{\text{rep}}N_W$. By this arrangement, each group handles the calculation of the correlation between the samples in the defined window. Each thread in a workgroup first reads a sample from a beamformed line, from each layer as shown in Fig. 2 and stores the sample value in shared memory. All threads within the workgroup now has fast access to samples read from other threads, via the shared memory, and almost 75% of the slow global memory reads are avoided.

Next, each thread in the workgroup computes a single entry in the cross-correlation table for each repetition, except for $n = N_{\text{rep}}$, and the maximum cross-correlation and corresponding delay index are found. To achieve subsample resolution of delay, the position of the $\arg\max_\zeta$ is refined using a parabola fit of the discrete samples around the index of maximum cross-correlation [10].

Finally, the median filter is implemented using a simple insertion sort.

III. Results

To evaluate the implementation, a measurement setup consisting of a flow circulation pump to control the flow velocity (Model: 75211-15, Cole-Parmer Instrument Co., Barrington) and a Danfoss MAGFLO flow meter for measuring the flow volume (Type MAG 3000, Dansk Fantom Service, Frederikssund, Denmark), was used. A BK Medical 10L2W linear array transducer was positioned at 62 deg relative to the flow direction and a BK Medical BK3000 scanner acquired and transferred data wirelessly to a HTC Nexus 9 tablet for imaging. Data was also stored for offline processing in Matlab for algorithm verification. One of the images created using Matlab is shown in Fig. 3 that illustrates a B-mode image overlayed with a CFM image.

Fig. 4 illustrates mean and ±1 standard deviation of 50 flow estimates of the central image line in the CFM image. The theoretical flow profile is shown in red. The relative bias is equal to $v_{bias} = 0.85\%$ and the relative standard deviation $v_{std} = 2.1\%$. This performance is comparable with the results in [7].

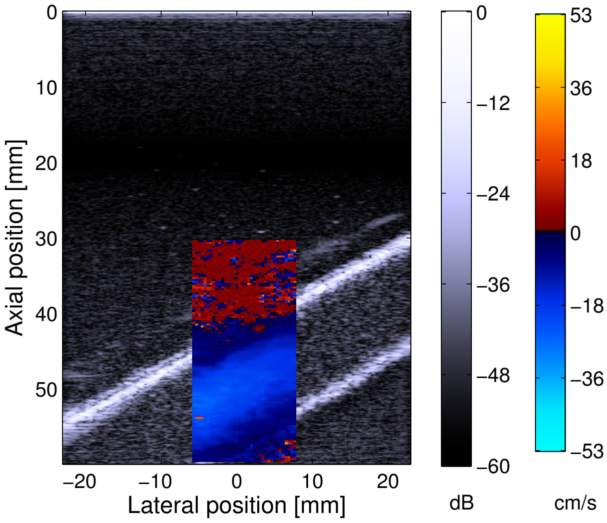

Fig. 3. B-mode image overlayed with CFM image. The tube is positioned at 62 deg relative to the flow direction. The CFM image shows the parabolic flow profile.

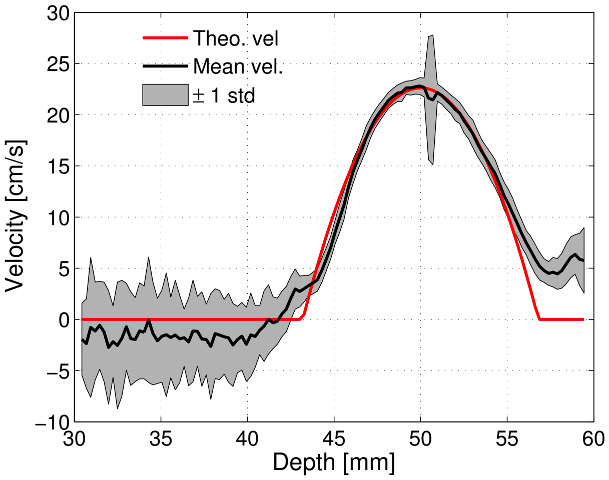

Fig. 4. Mean and ±1 standard deviation of 50 flow estimates of the central image line in the CFM image. The theoretical flow profile is shown in red. The large deviation observable at 50 mm is due to a noise spike that caused the cross-correlator estimator to choose a wrong lag.

A. Evaluation of the processing timing

Timing of the individual steps of the algorithm as detailed in Fig. 1 were acquired for the Nexus 9, which is one of the few mobile devices that supports the OpenGL ES 3.1 specification used for the implementation. As can be clearly seen in Fig. 5, most of the GPU-processing time is spent on the cross-correlation, which accounts for approximately 55% of the total amount of time used for a single frame. If general overhead from the data transfer and other android processes is not considered, nearly 65% is spent on the cross-correlation. The total time spent on both SASB and CFM with overhead is 25.7ms which is just below the acquisition rate of 29.4ms.

B. Evaluation of the wireless throughput

The data throughput is evaluated using an Asus RT-AC87U wireless router, which supports wireless 802.11ac technology

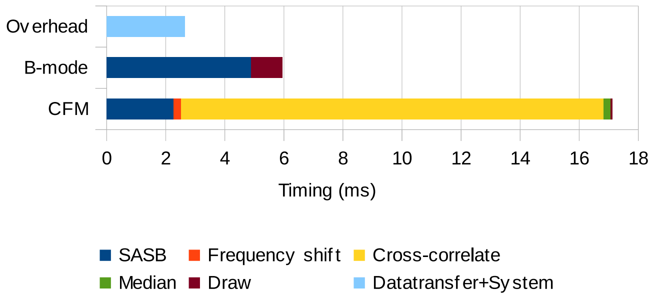

Fig. 5. Timings for the B-mode image (SASB) together with the individual steps for the CFM-image generation

and has 4 antennas, for a theoretical maximum transfer rate of 1733 Mbps on the 5 GHz band. The Nexus 9 has 2 embedded antennas and as such is only capable of receiving a theoretical maximum of 866 Mbps. Using zero message queue and sending data continuously from a computer wired to the 1 Gbit connection to the Nexus connected via the 5 GHz band, yielded an average result of 45 MB/s or 360 Mbps, which is above the required 36,7 MB/s needed for real-time processing.

C. *Evaluation of continuous processing*

As previously described, the high utilization of both GPU and Wi-Fi of the Nexus 9, was very close to the maximum performance of the tablet. All of the processing generates heat, which needs to dissipate through one small cooler attached to both GPU, CPU, and the network chip. At some point this causes the chip-set to enforce thermal throttling and real-time performance is no longer achievable. To try and pinpoint what causes this two test were made. In the first test, IQ-data was read from a file and in the second test data was received over Wi-Fi, but not processed in any wa on CPU or GPU. None of these experiments caused any performance degradation, which would have been expected, if either Wi-Fi or GPU utilization in isolation, cause the thermal throttling. This led to believe, that it is the combined heat generated from the network-chip, CPU and GPU. To test this the program was started with data throughput and the processing profile being measured over time as shown in Fig. 6. Data throughput starts out higher than the needed 36,7MB/s due to overhead from TCP/IP, ZMQ and WPA2 encryption. After two minutes the throughput and processing time falls below real-time requirements. After approximately six minutes without external cooling, performance has dropped down to roughly 24 frames per second. While running, the tablet was then placed on top of a large air ventilator and within three minutes full performance was again achieved, showing that heat from the entire system, is the probable cause of thermal throttling.

IV. Discussion and Conclusion

In conclusion, an implementation of a real-time duplex synthetic aperture imaging system on a commercially available hand-held device was presented. The implementation demonstrates that SASB can be used for both B-mode and CFM real-time ultrasound imaging, leading the way for ultra mobile ultrasound scanners with wireless probes. Other benefits such as continuous data and the ability to estimate the flow in a region at a time, compared to conventional techniques along scan lines, is also a major benefit of synthetic aperture flow. During the evaluation, new challenges arose. The thermal increase due to GPU, CPU, and network activity caused throttling and real-time performance could only be obtained for about two minutes.

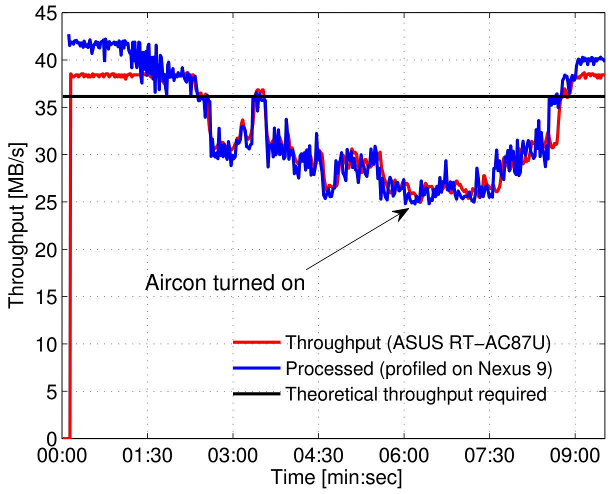

Fig. 6. Data throughput over time. External cooling turned on after approximately six minutes.

Acknowledgment

This work was supported by grant 82-2012-4 from the Danish National Advanced Technology Foundation and by BK Medical.

References

[1] GE, "GE Vscan," http://www.ge.com, 2014.

[2] Philips, "Philips Visiq," http://www.healthcare.philips.com, 2015.

[3] M. Hemmsen, P. M. Hansen, T. Lange, J. M. Hansen, K. L. Hansen, M. B. Nielsen, and J. A. Jensen, "In vivo evaluation of synthetic aperture sequential beamforming," *Ultrasound Med. Biol.*, vol. 38, no. 4, pp. 708–716, 2012.

[4] K. L. Gammelmark and J. A. Jensen, "Multielement synthetic transmit aperture imaging using temporal encoding," *IEEE Trans. Med. Imag.*, vol. 22, no. 4, pp. 552–563, 2003.

[5] S. I. Nikolov and J. A. Jensen, "In-vivo Synthetic Aperture Flow Imaging in Medical Ultrasound," *IEEE Trans. Ultrason., Ferroelec., Freq. Contr.*, vol. 50, no. 7, pp. 848–856, 2003.

[6] J. Kortbek, J. A. Jensen, and K. L. Gammelmark, "Sequential beamforming for synthetic aperture imaging," *Ultrasonics*, vol. 53, no. 1, pp. 1–16, 2013.

[7] Y. Li and J. Jensen, "Synthetic aperture flow imaging using dual stage beamforming: Simulations and experiments," *Journal of the Acoustical Society of America*, vol. 133, no. 4, pp. 2014–2024, 2013.

[8] M. C. Hemmsen, T. Kjeldsen, L. Lassen, C. Kjær, B. G. Tomov, J. Mosegaard, and J. A. Jensen, "Implementation of synthetic aperture imaging on a hand-held device," in *Proc. IEEE Ultrason. Symp.*, 2014, pp. 2177–2180.

[9] T. Kjeldsen, L. Lassen, M. C. Hemmsen, C. Kjær, B. G. Tomov, J. Mosegaard, and J. A. Jensen, "Synthetic aperture sequential beamforming implemented on multi-core platforms," in *Proc. IEEE Ultrason. Symp.*, 2014, pp. 2181–2184.

[10] S. G. Foster, "A pulsed ultrasonic flowmeter employing time domain methods," Ph.D. dissertation, Dept. Elec. Eng., University of Illinois, Urbana, Ill., 1985.