FPGA-based Real-Time Acoustic Camera Prototype

B. Zimmermann and C. Studer*

Integrated Systems Laboratory
ETH Zurich, 8092 Zurich, Switzerland
*e-mail: studer@iis.ee.ethz.ch

Abstract— **Acoustic cameras visualize the origin and intensity of sound waves using an array of microphones and sophisticated signal-processing algorithms. Due to the high memory-bandwidth and signal-processing complexity required by such algorithms, current available devices either compute the corresponding intensity-images off-line or require large and expensive hardware equipment. In this paper, we describe a low-complexity field-programmable gate array (FPGA)-based prototype, which computes and visualizes acoustic intensity-images in real-time. The system consists of 32 microphones and performs all signal processing tasks on a low-cost Xilinx Spartan 3E FPGA. The prototype computes intensity-images with a resolution of 320×240 pixels at 10 frames-per-second.**

I. Introduction

The visualization of origin and intensity of sound waves are well-established tools in ultrasound imaging [1], [2] and (underwater) sonar applications, e.g., [3]. The principle underlying such devices is to record audio signals from an array of microphones and to compute the intensity of sound pressure for well-defined points in space; this approach enables visualization of sound sources. For audible frequencies (i.e., in the range of about 20 Hz to 20 kHz), such devices are known as acoustic cameras (ACs) [4] and cover a large field of applications in practice. ACs can be used, e.g., for acoustic sound design of products and plants (e.g., noise reduction of airplanes), for detection and localization of defects in complex machines, or for design, planning of placement, and optimization of noise barriers in civil engineering. However, due to the large memory-bandwidth and high signal-processing complexity required by AC algorithms, most of the available solutions either perform the required signal processing algorithms off-line, e.g., [5], [6], or require large and expensive hardware equipment [4]. In contrast, affordable real-time solutions have been developed in the field of ultrasonics [7], which is mainly the case due to the large volume of devices sold; the associated signal-processing complexity is, however, still high.

We believe that almost all AC-applications would benefit from *real-time* visualization, as it provides the users with immediate feedback and hence, enables fast optimization of camera settings to different scenarios (e.g., position adjustment of the microphone array). In addition, real-time processing is imperative for applications, where the sound-source of interest only appears occasionally (e.g., during a short time interval) or at different locations during the measurement campaign. We emphasize that real-time ACs will not replace off-line processing and analysis, but rather complement the functionality of state-of-the-art off-line AC solutions.

Contributions: In this paper, we describe a low-complexity FPGA-based prototype implementation of a real-time AC. The system consists of 32 microphone modules with built-in pre-amplifiers, a 32-input A/D-conversion board, and a baseboard, which performs the required signal processing tasks on a low-cost FPGA and produces a real-time VGA output of acoustic intensity-images. The realized prototype system implements a reduced-complexity delay-and-sum beamforming algorithm, which requires low storage requirements and memory-bandwidth, and exhibits low signal-processing complexity. The employed FPGA architecture bases extensively on CORDIC (coordinate rotation digital computer) arithmetic, which enables real-time beamforming on low-cost FPGAs.

Outline: The remainder of this paper is organized as follows. Sec. II introduces the reduced-complexity delay-and-sum beamforming algorithm. Sec. III describes the key components of the prototype system. The FPGA architecture and corresponding implementation results are presented in Sec. IV. We conclude in Sec. V.

II. Delay-and-Sum Beamforming Algorithm

ACs visualize the origin and intensity of sound waves in a similar fashion as thermal cameras visualize origin and intensity of heat sources. To this end, a microphone array is focused consecutively to different discrete points in a well-defined area. Superposition of all microphone signals enables to compute estimates of the sound pressure for each selected spatial point, which translates to a pixel of the intensity-image. In our application, these points in space define a rectangular and planar grid in front of the array, where each point is assigned to a pixel in the final image. The calculated estimates of sound-pressure are then visualized by encoding sound-pressures to colors.

In order to focus the microphone array to a certain point in space, the array does not need to be adjusted physically. Instead, through summation of the signals picked up by all microphones with appropriate delays, one can focus the array to certain points in space, which is known as beamforming in the literature [8]–[10].

A. Delay-and-Sum Beamforming Algorithm

Fig. 1 illustrates the principle of delay-and-sum beamforming [9]. The key idea is to delay the signals from each microphone in such a way that all sound-waves originating from a particular point in space are in-phase and, therefore—when added together—interfere constructively. Sound-waves

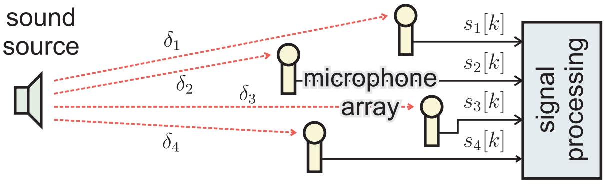

Fig. 1. AC principle: Sound waves arrive with delay δ_i to the ith microphone. Delayed combination of the audio-streams $s_i[k]$ enables to measure the intensity of the sound source through constructive interference.

originating from other points (than the selected one) in space are (up to a varying degree) out of phase and hence, interfere destructively. This approach enables to focus the microphone array to particular points in space by adjusting the delays of each microphone signal and summing these signals together.

The beamforming algorithm employed in our system is briefly summarized in the following. Each of the $i = 1, \ldots, N_{\text{mic}}$ microphones in the array generates a separate audio stream $s_i[k]$, where k stands for the sample index. The delay $\delta_i(\mathbf{p})$ between a certain point in space $\mathbf{p} = [\,p_x, p_y, p_z\,]$ and the ith microphone (cf. Fig. 1), with coordinates $\mathbf{m}_i = [\,m_x, m_y, m_z\,]$, is computed according to

$$\delta_i(\mathbf{p}) = \frac{f_{\text{s}}}{c} \|\mathbf{p} - \mathbf{m}_i\| \tag{1}$$

where the delays are measured in samples, f_{s} denotes the sampling frequency, $c \approx 340 \frac{\text{m}}{\text{s}}$ is the speed of sound, and $\|\cdot\|$ stands for the Euclidean norm. To achieve constructive interference, delayed versions of the N_{mic} microphone signals are summed together, i.e.,

$$s(\mathbf{p})[k] = \sum_{i=1}^{N_{\text{mic}}} s_i\big[k - \delta_i(\mathbf{p})\big]. \tag{2}$$

Note that interpolation can be employed, if $\delta_i(\mathbf{p})$ is not an integer; in our application, the signals are oversampled at the A/D converters by a factor of two and rounding towards the next integer is employed to reduce the computational complexity and the memory-bandwidth. The intensity of acoustic wave pressure at location $\mathbf{p}$ can finally be computed using (2) according to

$$I(\mathbf{p})[k] = \frac{1}{L} \sum_{\ell=0}^{L-1} \big|s(\mathbf{p})[k - \ell]\big|^2 \tag{3}$$

where $L > 0$ is an averaging parameter, which smoothens the intensity values over time, in order to reduce flickering artifacts in the resulting video signal.

B. Reduced-Complexity Beamforming

In order to enable real-time processing of delay-and-sum beamforming on low-cost FPGAs, two methods that reduce the memory bandwidth and storage requirements are described in the following paragraphs.

Reduction of memory-bandwidth: In order to compute intensity-images, the beam of the microphone array is steered to discrete points in an imaginary plane in front of the array in scan-line fashion. It can easily be seen that the algorithm requires numerous summations, squaring operations, and a large memory bandwidth (i.e., calculation of each pixel involves $L \cdot N_{\text{mic}}$ different audio-samples). The delays $\delta_i(\mathbf{p})$ in (1) can either be pre-calculated and stored in a memory or computed on-the-fly. Pre-calculation significantly increases memory bandwidth and storage requirements; the latter grows proportionally to the size of the image (e.g., an image of 320×240 pixels requires to store 2 457 600 delay values for an array of 32 microphones). In order to reduce memory bandwidth and storage requirements, we decided to compute the delays on-the-fly. This approach, however, requires a low-complexity and hardware-friendly way to calculate these delays (1) in hardware—a corresponding solution is described in Sec. IV.

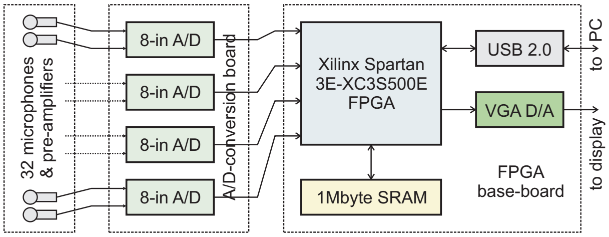

Fig. 2. System overview of the real-time AC prototype. Left: 32x microphones with pre-amplifiers; middle: 32-input A/D-converter board; right: FPGA base-board with video (VGA) output and USB 2.0 capability.

Reduction of storage requirements: If the distance from the microphone array to the plane to be scanned is large, the delay values in (1) are large as well, which substantially increases the required buffer sizes for storage of the audio samples $s_i[k]$. In order to reduce storage requirements, we only consider differences of delays to a reference point $\mathbf{m}_{\text{mid}}$, which is centered in the microphone array. Hence, instead of using (2) we compute the following sum

$$s(\mathbf{p})[k] = \sum_{i=1}^{N_{\text{mic}}} s_i\Big[k - \big(\delta_i(\mathbf{p}) - \delta_{\text{mid}}(\mathbf{p})\big)\Big] \tag{4}$$

where $\delta_{\text{mid}}(\mathbf{p}) = \frac{f_{\text{s}}}{c}\|\mathbf{p} - \mathbf{m}_{\text{mid}}\|$ corresponds to the delay from the center of the microphone array to a certain point $\mathbf{p}$ in space. This approach renders the storage requirements independent from the distance between the focal plane and the microphone array and hence, substantially reduces the amount of audio samples to be stored.

III. Real-Time Acoustic Camera System

The key components of the real-time AC prototype are illustrated in Fig. 2: The microphone array contains 32 microphones and includes 32 audio pre-amplifiers. A 32-input A/D-conversion board performs Nyquist filtering and A/D conversion. The FPGA base-board performs the reduced-complexity delay-and-sum beamforming algorithm in real-time and generates a corresponding video signal.

A. Microphone Array and Pre-Amplifiers

The microphone array consists of 32 microphone modules (of size 22 mm×11 mm), each containing an electret microphone (EMY-63M/P, Ekulit) with omni-directional characteristic. Rail-to-rail precision audio op-amps (LT1677, Linear

Technology Corporation) have been used for pre-amplification. The microphone modules are detachable from the A/D-conversion board, which allows for different microphone array geometries. Since pre-amplification is done right at the microphones, noise and interference can be kept at a minimum, especially if long cables are used (e.g., as it is required for large microphone arrays).

B. A/D-Conversion Board

The pre-amplified microphone signals are fed to the A/D-conversion board, which contains four 8-channel A/D-converters (CS5368, Cirrus Logic Inc.).[1] Each A/D converter supports a maximum sampling frequency of 216 kHz at a resolution of 24 bit. In the current application, a sampling rate of only 85.9 kHz is used, which significantly reduces the amount of data to be processed and is easily generated from the 66 MHz main clock oscillator (i.e., only requires division of the clock signal by 768). All necessary input buffers and anti-aliasing filters for each of the 32 channels are realized by dual op-amps (TS922, ST Microelectronics). The connections between microphones and the A/D-conversion board are placed in a 4×8 grid, which leads to a rectangular array that can be used for test purposes. The 32 sampled audio streams are transmitted to the FPGA base-board using time-division multiplexing (TDM). Eight channels are transmitted serially over one line at a data rate of approximately 16 Mbit/s.

C. FPGA Base-Board

The FPGA base-board performs the reduced-complexity delay-and-sum beamforming algorithm described in Sec. II in real-time and generates the video signal of the computed intensity-images. A low-cost Xilinx Spartan 3E XC3S500E FPGA running at a clock frequency of 66 MHz has been used. In addition to the TDM ports required for the digitized audio signals, a video graphics array (VGA) video DAC (ADV7125, Analog Devices Inc.) is attached to the FPGA through its general-purpose I/O ports. The FPGA base-board additionally contains stereo audio output, which enables to listen to a specified point in space (i.e., the system can be used as a directional microphone as well). Furthermore, USB 2.0 capability has been implemented to enable data transfer to a PC and for configuration of the AC.

IV. Architecture and Implementation Results

In this section, the FPGA architecture of the beamforming algorithm is described and implementation results of the real-time AC prototype system are reported.

A. FPGA Architecture Overview

Fig. 3 provides an overview of the AC architecture. The TDM sample streams generated by the four A/D-converters are de-multiplexed and written to 32 ring-buffers. These buffers are implemented in the BRAM modules provided by the FPGA. Since the FPGA does not offer enough BRAM units to dedicate one instance to each ring-buffer, 16 BRAMs run at twice the clock frequency to create 32 virtual ring-buffers. These virtual ring-buffers have half the storage capacity of the original BRAMs and provide storage for 256 audio samples.

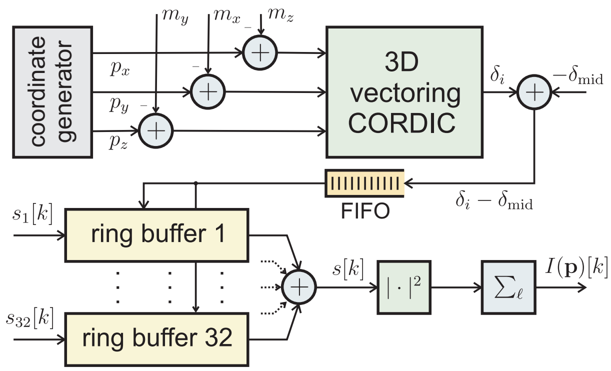

Fig. 3. Real-time AC architecture overview. The 3D vectoring CORDIC computes delays which are used to read-out samples from the ring buffers.

As mentioned in Sec. II-B, the necessary delays associated with each scanned point in space are computed on-the-fly. These delays are computed in three-dimensional (3D) vectoring CORDIC (see Sec. IV-B for more details), which obtains (pre-calculated) coordinates of each microphone and each generated pixel-location in the focal plane. The calculated offsets $\delta_i - \delta_{\mathrm{mid}}$ are then stored in a FIFO for rapid access.

Through the secondary (fully-independent) ports of the BRAMs, the samples in the ring buffers are retrieved according to the calculated offsets and summed together by a pipelined adder tree. The signal is then high-pass filtered (not shown in Fig. 3), squared, and smoothened over $L = 70$ samples. The resulting intensity values $I(\mathbf{p})[k]$ are written to a dual-buffered video memory, which is realized in the 1 Mbyte SRAM on the FPGA base-board. An independent unit manages the data in the video memory, i.e., reads data from the video buffer, generates corresponding intensity-values according to a pre-defined color-map, and outputs these values to the video DAC.

B. High-Throughput 3D Vectoring CORDIC

In order to calculate the delays $\delta_i(\mathbf{p})$ between a given point in space $\mathbf{p}$ and the ith microphone location $\mathbf{m}_i$, vectoring CORDICs [11] have been used. This approach is known to efficiently compute Euclidean distances of two-dimensional vectors in hardware. In our application, however, distances of 3D vectors are required (see Eq. 1). Computation of the Euclidean distance of a 3D vector $\mathbf{v}$ is performed by cascading two vectoring CORDICs in such a way, that the first instance obtains v_x and v_y and its output (which corresponds to $\sqrt{v_x^2 + v_y^2}$) is connected to one input of the second CORDIC. By feeding v_z to the other input, the output of the second CORDIC corresponds to $\|\mathbf{v}\| = \sqrt{v_x^2 + v_y^2 + v_z^2}$, which is exactly what is needed to compute (1).

Each CORDIC performs a total of 20 micro-rotations and has an internal precision of 16 bit. In order to achieve high throughput, both CORDICs are fully pipelined (i.e., one pipeline stage per micro-rotation is used). The resulting

[1] Due to stringent limitations of the available block-RAM (BRAM) slices and the used low-cost FPGA, the signal-processing complexity for 32 channels is already at the maximum possible with the current AC-prototype.

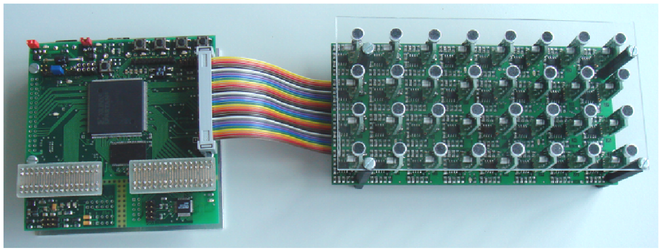

Fig. 4. Complete real-time AC prototype. Left: FPGA base-board; right: 32 microphone modules directly attached to the A/D-conversion board.

architecture delivers one Euclidean distance per clock cycle, ultimately leading to a maximum of 66 million delay computations per second (as both CORDICs are running at 66 MHz).

C. Implementation Results

FPGA implementation results: The final FPGA implementation results are provided in Tbl. I. In light of the low-cost FPGA that has been used to implement the real-time AC prototype, the slice and BRAM utilization of 81% and 95%, respectively, demonstrates the low-complexity of the implemented beamforming algorithm. Note that this is a result of reducing the complexity of the underlying algorithm as shown in Sec. II-B and using a fully pipelined 3D vectoring CORDIC for on-the-fly delay calculation (see Sec. IV-B).

Prototype system: Fig. 4 shows the real-time AC prototype, consisting of the FPGA base-board, the A/D-conversion board, and the microphone array. The microphone modules are plugged directly into the A/D-conversion board to form a 4×8 rectangular array with a distance d of 19 mm between adjacent microphones. A glass plate ensures correct spacing of the microphones. This array configuration leads to a (spatial) Nyquist frequency of about $f_{\lambda/2} = \frac{c}{d} = \frac{340\,\frac{\mathrm{m}}{\mathrm{s}}}{0.019\mathrm{m}} = 17.9\,\mathrm{kHz}$ and signals above this frequency cause strong aliasing artifacts. The small size of the prototype allows for a simple integration into portable applications. The AC has been tested in different settings using various sound sources and was shown to be fully functional. Fig. 5 shows a typical AC picture for the rectangular microphone array depicted in Fig. 4. The key specifications of the implemented real-time AC are summarized in Tbl. II.

TABLE I

Implementation Results on Xilinx Spartan 3E (XC3S500E)

Clock frequency	66 MHz
Slices	3771 (81%)
Slice flip-flops	5587 (60%)
LUT's occupied	4656 (50%)
Block-RAMs	19 (95%)

TABLE II

Real-Time Acoustic Camera Specifications

Microphones	32
Sampling rate	85.9 kHz
Resolution	24 bit
Frequency response	600 Hz–42 kHz
Video resolution	320×240
Intensity levels	256
Frame rate	10 fps

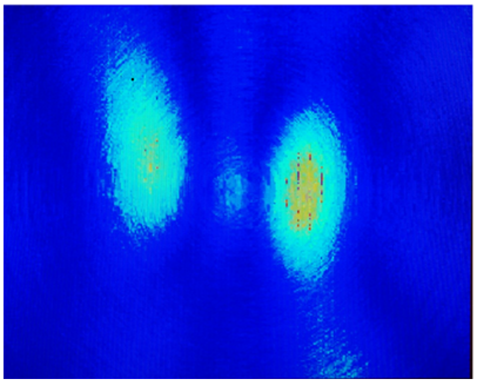

Fig. 5. Typical output image of the real-time AC prototype. Two loudspeakers generate 10 kHz sine-waves and are placed approximately two meters in front of the 4×8 rectangular microphone array.

V. Conclusion

In this paper, a real-time AC prototype has been described. Signal processing is performed on a FPGA using a reduced-complexity delay-and-sum beamforming algorithm. The corresponding architecture bases on CORDIC arithmetic. Our final implementation achieves 10 fps for 320×240 pixels, which demonstrates that acoustic intensity-images can be calculated in real-time even on low-cost FPGAs. The modularity of the system allows for different array geometries and enables to conduct real-world experiments with the current platform. Thanks to its small size, the prototype system can be used for portable applications. In near future, we plan to use a more powerful FPGA, to implement more sophisticated signal processing algorithms that provide better resolution (in terms of pixels and intensity levels) and higher frame-rates.

Acknowledgments

The authors would like to thank D. Perels, N. Felber, H.-P. Mathys, H.-J. Gisler, R. Köppel, and W. Fichtner for their support during the realization of the AC prototype.

References

[1] O. T. V. Ramm and S. W. Smith, "Three-dimensional imaging system," US Patent 4 694 434, 1987.

[2] S. H. Maslak and J. N. Wright, "Phased array acoustic imaging system," US Patent 4 550 607, 1985.

[3] R. J. Urick, *Principles of Underwater Sound*, 3rd ed. New York: McGraw-Hill, 1983.

[4] gfai tech GmbH, 12489 Berlin, Germany, "Acoustic camera: listening with your eyes," `http://www.acoustic-camera.com`.

[5] LMS International, 3001 Leuven, Belgium, "LMS test.lab high definition acoustic camera," `http://www.lmsintl.com/testing/testlab/acoustics/high-definition-acoustic-camera`.

[6] Microflown Technologies, 6900 AH Zevenaar, The Netherlands, "Charting sound fields," `http://www.microflown.com`.

[7] C.-H. Hu, X.-C. Xu, J. M. Cannata, J. T. Yen, and K. K. Shung, "Development of a real-time, high-frequency ultrasound digital beamformer for high-frequency linear array transducers," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, pp. 317–323, Feb. 2006.

[8] B. D. V. Veen and K. M. Buckley, "Beamforming: A versatile approach to spatial filtering," *IEEE ASSP Magazine*, pp. 4–24, Apr. 1988.

[9] M. Brandstein and D. Ward, Eds., *Microphone Arrays: Signal Processing Techniques and Applications*. Springer, 2001.

[10] H. Krim and M. Viberg, "Two decades of array signal processing research," *IEEE Sig. Proc. Magazine*, pp. 67–94, Jul. 1969.

[11] B. Parhami, *Computer Arithmetic Algorithms and Hardware Designs*. Oxford Univ. Press, New York, 2000.