# Steerable distributed large-aperture audio array using low-power wireless acoustic sensor nodes

János Sallai, Ákos Lédeczi, Xenofon Koutsoukos, and Péter Volgyesi

Vanderbilt University
2015 Terrace Place, Nashville, TN, 37203, USA
janos.sallai@vanderbilt.edu

**Abstract.** Recent advancements in low-power wireless communication technologies allow for rapid deployable embedded monitoring applications with a deployment lifetime of several weeks of months. In this work, we present a work-in-progress wireless audio streaming solution that uses a network of small wireless microphone arrays to form a distributed steerable large-aperture array. To keep the application's communication bandwidth requirements manageable, we use a hierarchical data fusion algorithm that allows for delay-and-sum style beamforming to be carried out in-network, streaming the audio data along a routing tree to a base station. With the proposed distributed beamforming approach, it is sufficient to communicate as few as $log(n)$ data streams for a network of $n$ sensors, which is a significant reduction from a centralized approach that would require all $n$ streams to be routed to the base station.[1]

**Key words:** wireless communication, acoustic beamforming, in-network aggregation

## 1 Introduction

Beamforming with distributed microphone arrays has been a widely applied acoustic sensing technique for several decades [3]. The overwhelming majority of existing applications in the realm of surveillance and acoustic event detection, however, have been line powered and have relied on wired communications [6, 5, 7]. With the emergence of low power wireless hardware and the corresponding communications protocols, new application possibilities are emerging, offering reduced deployment time, flexible deployment topology and battery powered operation for several weeks or months.

Let us consider the following surveillance application scenario. A public, potentially crowded area (café, railway station, stadium, etc.) is instrumented with a number of wireless microphone arrays. A special agent intends to eavesdrop on a conversation between a small group of people in the crowd. Given the coordinates the point of interest (i.e. the physical location of the group), the special

[1] This work was supported part by the ARO MURI grant W911NF-06-1-0076.

agent steers the distributed sensor array to that location, which in turn amplifies the sound originating there, suppresses the background noise (wind, traffic noise, human speech coming from other speakers), and forwards the audio stream to the special agent.

In this work, we propose a method to implement a family of beamforming algorithms that satisfy certain properties in a distributed fashion. The proposed distributed computation allows for reducing the overall communication needs of the distributed beamforming system without loss of information.

In contrast to the conventional approach to implementing the above described system, which involves wired microphones that are directly connected to an embedded computer that carries out the beamforming operation, our research focuses on fitting the distributed beamforming application to a wireless sensor network. Casting the problem as a sensor network problem, we propose an in-network data fusion technique that reduces the communication requirements of the application by distributing the delay-and-sum operation. This is carried out by aggregating the audio streams from the individual multiple-microphone nodes along a multi-hop routing tree: the user at the root of the tree will receive just the amplified stream, eliminating the need for directing all microphone streams to central processing location.

We present a work-in-progress sensor platform, featuring eight microphones with eight dedicated high-speed ADC channels, FPGA based processing that allows for local beamforming computation and 802.15.4 compliant radio module for low-power wireless networking. We anticipate that the end system will be able to process and stream human speech with 802.15.4/ZigBee class low power radios.

## 2 Aggregation

In-network data fusion, i.e. processing audio streams originating from the microphones within the sensor network, can result in significant gains in terms of communication bandwidth requirements, computational complexity and power consumption. In this work, we show that delay-and-sum beamforming [1] can be computed in a parallel and distributed fashion in a computational tree. This tree can coincide with the routing tree the leaves of which are the acoustic sensors, and the root of which is the user where the acoustic stream is consumed. Also, we show that through in-network aggregation, communication bandwidth requirements can be reduced to order of $log(n)$ for $n$ microphones, which is a significant decrease compared to order of $n$ for a centralized solution. Let us consider delay-and-sum beamforming, one of the simplest beamforming techniques. Given a set of spatially distributed microphones at known positions sampling a broadband signal emitted by a source at a given location. The microphone array is steered to the given location by selecting appropriated phases for each microphone, such that the streams from each microphone $x_i(t)$ are delayed with the time it takes for the signal to travel from the source to the respective microphones. The streams are (optionally) weighted and then summed sample wise,

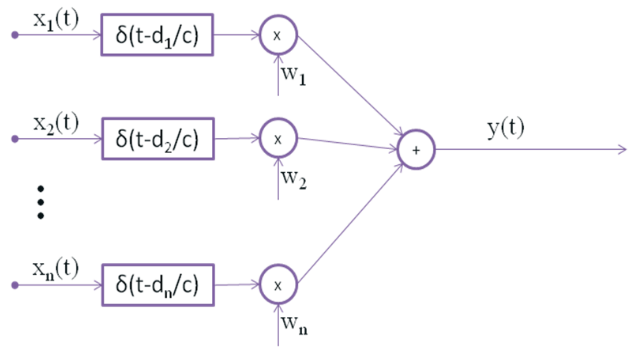


**Fig. 1.** Delay-and-sum beamforming.

resulting in the signal

$$y(t) = \sum_{i=1}^{n} w_i x_i(t - d_i/c) \tag{1}$$

which, for the signal originating from the point of interest, has a gain of $\sum w_i$ and signal to noise ration of $\frac{Pn}{\sigma^2}$, where $P$ is the power and $\sigma^2$ is the noise variance of the signal at the respective microphones.

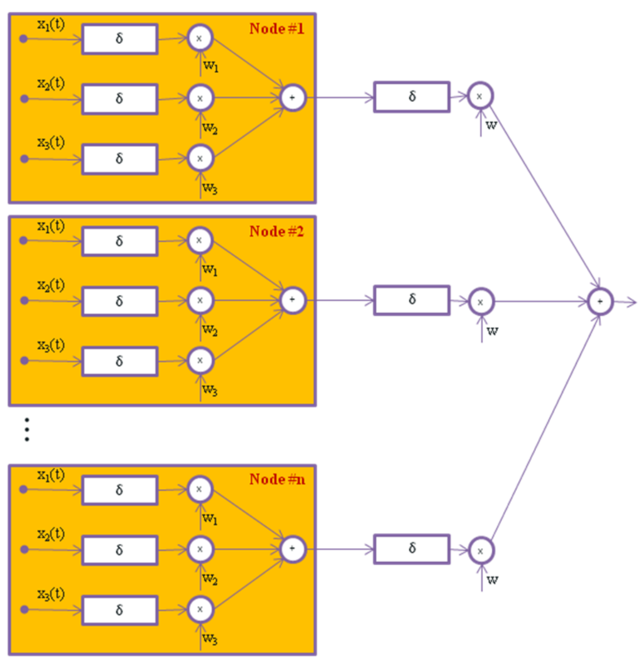


**Fig. 2.** Hierarchical decomposition of delay-and-sum computation.

A key property of the delay and sum operation is that it allows for hierarchical decomposition. The order of operations does not matter (associativity): it is possible to align a subset of the streams with appropriate delays, compute the sum, and propagate the aggregate stream up in a hierarchy without altering the end result. This allows us to distribute the computation in the network, and to restrict node-to-node communication to aggregate streams. Assuming a tree topology, this technique can reduce communication requirements from requiring to stream all $n$ microphones to a centralized location to communicating $log(n)$ aggregate streams along the edges of a routing tree.

Specifically, the audio streams received by the eight local microphones are fused locally at the node. The aggregate stream is compressed and communicated down a routing tree toward the user at the root of the network. At every in-node of the routing tree, the multiple incoming streams are fused (summed with the appropriate delays), and the resulting single stream is propagated downwards. At the root of the tree, the resulting single stream is then normalized and provided to the user.

Logically, therefore, there are three different operations (using a distributed databases terminology [4]) on the streams that are carried out by the application.

**Initializer.** The initializer function, running at the leaf nodes, maps sensor readings to a partial computational result (PCR). The PCR is defined as a three-tuple, consisting of a stream of samples $[x_0, x_1, ..., x_k]$, the time of emission of the first sample $t$, and the weight $w$ associated with the stream. It is the partial computational result that is communicated in the network. The initializer function is the following:

$$i([x_0, x_1, ..., x_k]) = \langle [x_0, x_1, ..., x_k], t_0 - \frac{d}{c}, 1 \rangle$$

That is, it wraps the samples in a PCR, timestamps it with the time of acquiring the first sample ($t_0$) minus the time-of-flight of the signal ($d$ is the distance between the sound source and the microphone, while $c$ is the speed of sound), and assigns it the weight 1.

**Merging function.** The merging function, running at the in-nodes of the tree, maps two input PCRs to an output PCR by aligning the two input streams according to their delays, and tagging the result it with the sum of the weights of the inputs:

$$f(\langle [x_{1,0}, x_{1,1}, ..., x_{1,k}], t_1, w_1 \rangle, \langle [x_{2,0}, x_{2,1}, ..., x_{2,k}], t_2, w_2 \rangle) =$$
$$\begin{cases} \langle [x_{1,0}, x_{1,1}, ..., x_{1,k}] + \delta([x_{2,0}, x_{2,1}, ..., x_{2,k}], t_2 - t_1), t_2 - t_1, w_1 + w_2 \rangle \; if t_1 < t_2 \\ \langle \delta([x_{1,0}, x_{1,1}, ..., x_{1,k}], t_1 - t_2) + [x_{2,0}, x_{2,1}, ..., x_{2,k}], t_1 - t_2, w_1 + w_2 \rangle \; if t_1 > t_2 \end{cases}$$

The merging function is associative, that is, it satisfies the equation:

$$f(f(\langle [x_{1,0}, ..., x_{1,k}], t_1, w_1 \rangle, \langle [x_{2,0}, ..., x_{2,k}], t_2, w_2 \rangle), \langle [x_{3,0}, ..., x_{3,k}], t_3, w_3 \rangle) =$$
$$f(\langle [x_{1,0}, ..., x_{1,k}], t_1, w_1 \rangle, f(\langle [x_{2,0}, ..., x_{2,k}], t_2, w_2 \rangle, \langle [x_{3,0}, ..., x_{3,k}], t_3, w_3 \rangle))$$

**Evaluator.** The evaluator function maps an input PCR to a final result stream. In particular, it divides the samples in the input with the weight as follows:

$$e(\langle [x_0, ..., x_k], t, w \rangle) = \frac{1}{w}[x_0, ..., x_k]$$

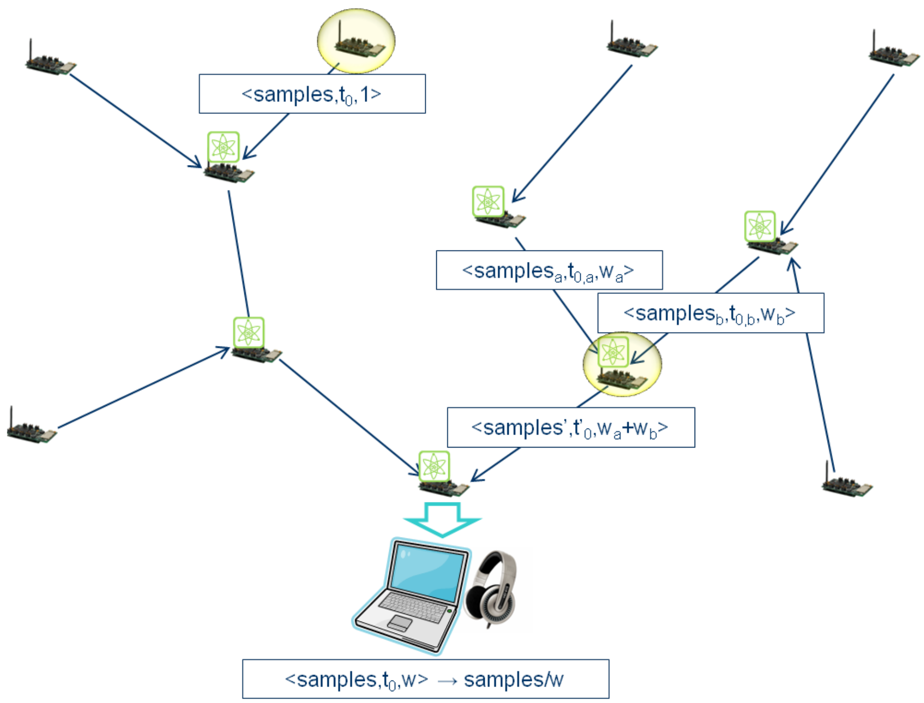


**Fig. 3.** In-network computation.

The distributed beamforming system, as shown in Figure 3, will operate as follows. Each internal node of the network will recursively merge incoming partial results and output a partial result, by applying the merging function. This process is bootstrapped at the leaf nodes, where timestamped samples from the microphones are wrapped into partial results initially, by applying the initializer function. In the end, the partial result at the root of the routing tree is converted to the final result of the beamforming computation by applying the evaluator function.

Notice that this method avoids communicating all $n$ streams (where $n$ is the number of microphones) to the root (or any other central processing location). It requires that only $log(n)$ aggregate streams are communicated along the edges of a routing tree, which results in a considerable reduction in energy and bandwidth, making this technique particularly suitable for wireless applications.

## 3 Hardware

In order to experimentally validate the feasibility of the proposed distributed beamforming approach, we are developing a wireless microphone array platform that will host the beamforming application. In this section, we describe the specifics of the hardware that is in production phase at the moment.

Our sensor hardware features eight ADC channels that can host up to eight high sensitivity microphones per array. The node is built around a Xilinx Spartan 3 FPGA that hosts the signal processing tasks and handles compression and decompression of data streams for bandwidth-efficient inter-node communication. FPGAs are particularly suitable for processing multichannel high data rate sample streams, because they allow for parallelizing most of the signal processing tasks, therefore they are faster and more power efficient for these specific tasks than traditional microcontrollers or DSPs. Communication is handled by an on-board ZigBit wireless sensor network module, built around an ATMega1281 low-power microcontroller and an 802.15.4 compliant radio transceiver, which allows for reliable, standards compliant low-power communication between network nodes. In addition to the microphones, the hardware platform also features a GPS module and a 3-axis accelerometer, which will be used in future applications.

The ZigBit module, configured for low-power listening communication (using the radio MAC layer of TinyOS 2.x), can draw as little as a 50uA at 3V on average, therefore this is the component we selected to handle the power management tasks in our platform. The low power listening enabled radio stack keeps the radio turned off most of the time and wakes up occasionally to check if there is communication on the channel. The FPGA is turned off by default, and is woken up on-demand by the ZigBit module. A wake-up event can be either a radio message received by the node, or an acoustic event detected by the wake-up circuitry of the first ADC channel. The acoustic wake-up circuitry allows for detecting a high-energy acoustic event using an analogue circuitry, while keeping the relatively power-hungry FPGA in power off when not in use.

## 4 Conclusion

In this paper, we presented an approach, which is novel to our knowledge, to distribute the computation in-network of a class of beamforming algorithms. The proposed method offers a reduction in the number of streams to be communicated to $log(n)$ (where $n$ is the number of microphones) from $n$ streams in the traditional non-distributed approach. We expect that such savings in communication requirements will make this technique applicable for wireless distributed acoustic arrays using low-power radios, with the potential of several days of deployment lifetime in active streaming operation mode, and several weeks or months in standby. While in Section 2 the proposed approach is explained assuming delay-an-sum beamforming, it is, in fact, applicable to more

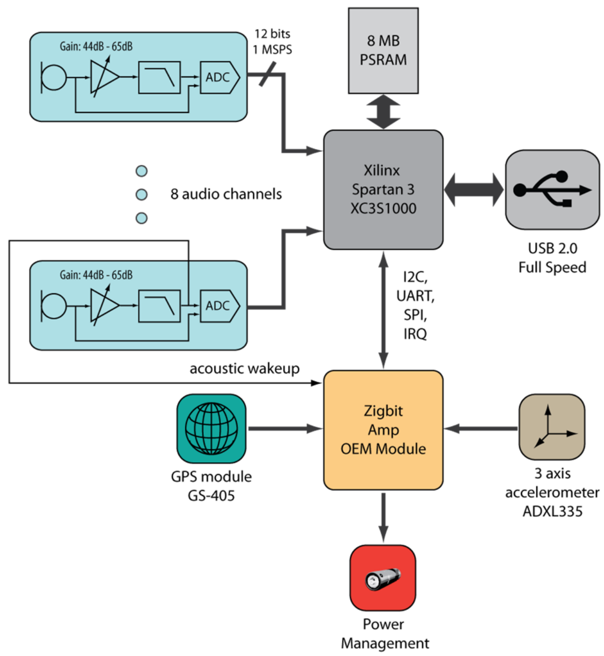


**Fig. 4.** Sensor node hardware.

sophisticated beamforming techniques (e.g. polynomial or filter-and-sum beamforming [2]) that allow for hierarchical decomposition with a suitable associative merging function.

We continue this work with experimentally validating the distributed beamforming approach presented in this paper. Currently, the hardware platform that will host the proof-of-concept application is in production phase. The next steps include developing the firmware for the FPGA, including ADC drivers, MCU interface, digital filtering, delay-and-sum beamforming and compression, in parallel with the MCU code for the ZigBit module to facilitate tree formation and communication. The experimental evaluation will include quantitative benchmarking with high-sensitivity microphones from different vendors, and will provide further information on the range and attainable noise levels of individual nodes as well as the entire steerable distributed acoustic array.

## References


1. R. P. Dougherty. What is beamforming? BeBeC-2008-01, 2008. Proceedings on CD of the 2nd Berlin Beamforming Conference, 19-20 February, 2008.
2. M. Kajala and M. Hamalainen. Filter-and-sum beamformer with adjustable filter characteristics. In *ICASSP '01: Proceedings of the Acoustics, Speech, and Signal Processing, 2001. on IEEE International Conference*, pages 2917–2920, Washington, DC, USA, 2001. IEEE Computer Society.

3. H. Krim and M. Viberg. Two decades of array signal processing research: the parametric approach. *Signal Processing Magazine, IEEE*, 13(4):67 –94, jul 1996.
4. S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tag: a tiny aggregation service for ad-hoc sensor networks. *SIGOPS Oper. Syst. Rev.*, 36(SI):131–146, 2002.
5. H. Silverman, I. Patterson, W.R., and J. Flanagan. The huge microphone array. *Concurrency, IEEE*, 6(4):36 –46, oct-dec 1998.
6. SquareHead technologies website. `http://www.squarehead.com`.
7. E. Weinstein, K. Sttele, A. Agarwak, and J. Glass. Loud: A 1020-node microphone array and acoustic beamformer. *International Congress on sound and Vibration (ICSV), Cairns, Australia*, July 2007.