Real-Time Independent Component Analysis Implementation and Applications

Marcos Turqueti, Jafar Saniie and Erdal Oruklu

***Abstract*—A common problem in disciplines such as high energy physics, biomedicine and acoustic signal processing is finding a suitable representation of multivariate data. Independent Component Analysis (ICA) is a recently developed mathematical tool that can recover independent source signals and is now mature enough to be implemented in real-time applications such as photomultipliers signal processing, magnetic resonance imaging and acoustic arrays. This technique is based on the assumption that signals from different sources are statistically independent and statistically independent signals can be extracted from mixture signals. ICA defines a model for the observed data that requires a large number of samples in order to establish the necessary statistics. The model assumes that the data variables are linear combination of unknown variables, the unknown variables are assumed to be non-Gaussian and independent. The goal then becomes to find a transformation in which the components are as statistical independent as possible from each other. This technique is related with methods such as principal component analysis and factor analysis. The ICA algorithm is computing intensive since it must accumulate and go through the signal samples performing complex operations. Efficient versions of the algorithm have being already deployed using different techniques such as the FastICA that can be implemented efficiently in hardware platforms such as DSP processors and FPGA's. In this paper, we present the ICA principles, implementation and current applications.**

I. Introduction

THIS paper describes the implementation of a system based on Independent Component Analysis dedicated to solve the problem of blind source separation. This system utilizes specially design hardware, configware and software to achieve real-time source separation.

In order to demonstrate the system capabilities towards source separation, the system is coupled with an acoustic array to acquire physical signals and demonstrate its capabilities operating with real signals.

Manuscript received June 4, 2010.

Marcos Turqueti is with the Illinois Institute of Technology, Chicago,IL, 60616 USA (telephone: 630-840-2244, e-mail: mturquet@iit.edu).

Jafar Saniie is with the Illinois Institute of Technology, Chicago,IL, 60616 USA (telephone: 312-567-3412, e-mail: sansonic@ece.iit.edu).

Erdal Oruklu is with the Illinois Institute of Technology, Chicago,IL, 60616 USA (telephone: 312.567.8814, e-mail: erdal@ece.iit.edu).

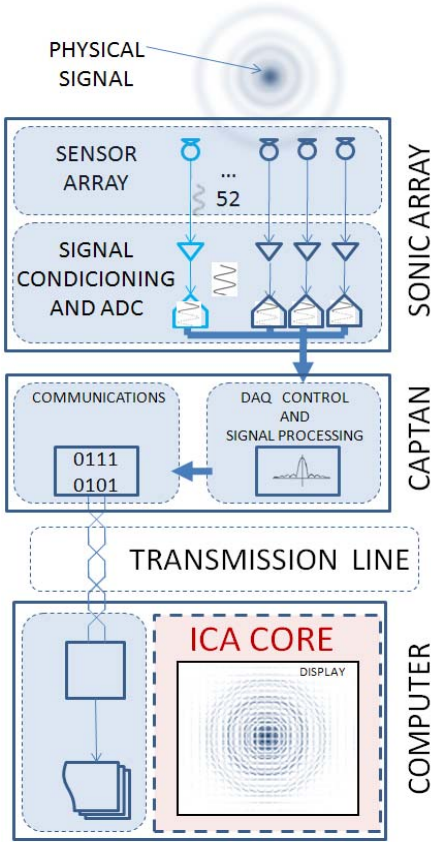

Fig. 1. The three main components of the system from top to bottom; sonic array, CAPTAN and computer.

As integral part of this system we have design an acoustic MEMS (Microelectromechanical systems) array, the electronics of the array have demanding issues to deal with like: electronic noise, power decoupling, cross-talk, and connectivity. Furthermore, the system needs to integrate the acoustic array with real-time data acquisition, signal processing, and network communication capabilities. In this type of system, characteristics such as speed, scalability, and real-time signal processing are paramount and demand highly advanced data acquisition and processing modules. The

978-1-4244-7110-2/10/$26.00 ©2010 IEEE

Authorized licensed use limited to: Illinois Institute of Technology. Downloaded on October 02,2020 at 06:04:11 UTC from IEEE Xplore. Restrictions apply.

acoustic array was integrated with a data acquisition system design for high energy physics applications called CAPTAN (Compact And Programmable daTa Acquisition Node) [1]. The objective of this work is to present the system above mentioned operating in conjunction with the ICA algorithm performing sound source separation.

II. Acoustic Array

The design of the acoustic array board was based on three basic requirements; good spatial resolution, high signal to noise rate (SNR) and user selectable unidirectional/omni-directional acoustic aperture (i.e., beam steering). Spatial resolution is governed by the number of MEMS microphones, inter-microphone distance, and microphone sensitivity. This acoustic array is made of 52 microphones; this number was chosen in order to obtain a highly flexible system with good spatial resolution and sensitivity, and high SNR. The microphones are distributed in an octagonal grid with the inter-microphone distance of 10.0 mm centre to centre in the horizontal and vertical. The spatial sampling rate of this array can be approximated by dividing the sound speed by the inter-microphone distance and then further dividing the result by two in order to satisfy the Nyquist-Shannon theorem [2]. Spatial sampling rate of frequencies up to 18 kHz without spatial aliasing can be achieved with this spacing.

The sensitivity of the array increases monotonically with the number of sensors, and the MEMs microphone chosen for this array (SPM0208) has a sensitivity of 1V/Pa at 1 kHz [3]. These microphones are Omni-directional and when combined in the array they provide a very good acoustic aperture.

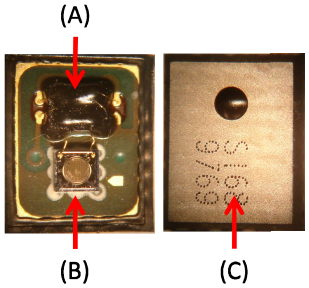

Fig. 2. MEMS microphone, (A) amplifier, (B) microphone and (C) aluminium cover.

The MEMs microphones are a fundamental piece of the array due to the small size, high sensitivity, and low reverberation. Its frequency response is essentially flat from 1 to 10 KHz and it has a low limit of 100Hz and a high limit of 25 KHz. The array geometry was chosen in order to allow beam steering on the horizontal and vertical planes [4].

In order to reduce reverberation on the system the microphones are glued with silver epoxy to the surface mount copper pads in the board. The array also provides a central loudspeaker that has dual use: it can be used for calibration purposes, or for sonar like applications. When used as calibration element the microphone emits a set of pure frequencies that then are captured by the microphones and used for its calibration taking in account the geometry of the array.

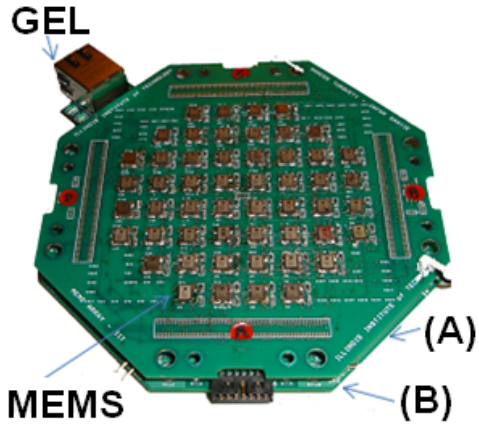

Fig. 3. Top view of the acoustic array. The MEMS elements are clearly visible on the top of the printed circuit board (PCB).

When the central microphone is used for active sonar applications a series of pre-programmed pulses is generated and captured back by the array.

In support of the microphones the board also provides amplification and analog-to-digital conversion (ADC). Every single channel has two operational amplifiers, one embedded in the MEMS package as illustrated by Fig. 2, and one in the back of the PCB board. An individual ADC is provided for every channel, each ADC has 12 bits resolution, 2.0 Volts dynamic range and a maximum sampling rate of 5Mbps.

Due to the nature of the array, this circuit board requires extensive use of decoupling capacitors, there is a total of 54 10uF capacitors on the back of the printed circuit board, one for every channel, this proofs to be fundamental since all the channels are parallel and will essentially have a peak demand of power at the same time. The second stage amplifier provides a first order high-pass filter with corner frequency at 400Hz.

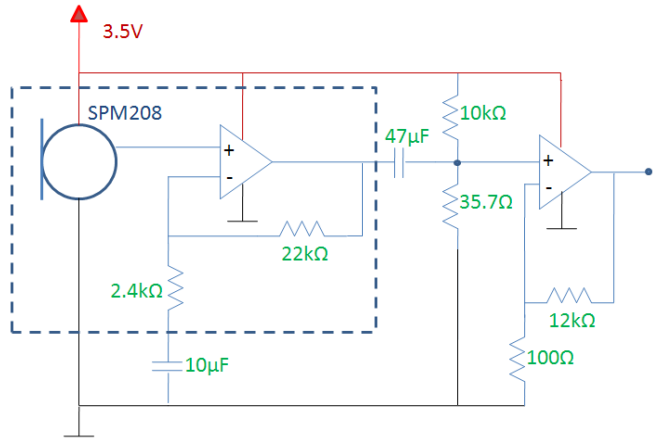

Fig. 4. Electronics circuit supporting each individual channel.

Fig. *4* provides a high level schematic of the analog supporting circuitry for each channel. The first stage amplifier gives a 20dB gain on the signal and it is AC coupled to the second stage that gives additional 20dB of

Authorized licensed use limited to: Illinois Institute of Technology. Downloaded on October 02,2020 at 06:04:11 UTC from IEEE Xplore. Restrictions apply.

gain. After the signal is digitized it is passed to the digital board called Node Processing and Control board (NPCB) [2]. The NPCB is stacked underneath the array.

III. Readout and Processing System

The digital part of the sonic array system is implemented in the NPCB. The NPCB board is part of the CAPTAN system, the CAPTAN system is a distributed architecture based on core elements known as system nodes. A node is a stack of boards connected together by the Vertical Bus in which every board in the same node has access to the Vertical Bus and is therefore accessible to each other.

Fig. 5. Top view of the NPCB board. In the centre the FPGA, surrounding it the four vertical bus connectors and in the right the Gigabit Ethernet card.

Basically, the NPCB board contains a VIRTEX-4 XC4VFX12 FPGA [6], an Ethernet Gigabit link and an EPROM. All data from the acoustic array is stored and processed on this board. The FPGA chip is booted by the 32MB EPROM that contains a specially designed configware for managing the data coming from the sonic array.

There are in total 68 digital lines interconnecting the sonic array and the NPCB, 52 data lines from the channels, 8 clock lines, and 8 data valid. These lines are evenly distributed into the four board-to-board Vertical Bus connectors that the system provides, 17 LVCMOS lines per connector. The power is also provide through these connectors, where the system is powered in a single 3.3V connection to the NPCB board, regulated and then distributed to the sonic array.

All the lines on the NPCB board have the length matched to avoid timing issues and routed in alternated copper layers between power planes to reduce cross-talk. The clock is distribution in a star fashion to the ADCs on the sonic array; each clock line can support 7 or 6 ADCs depending on the quadrant of the board.

This board also provides a Gigabit Ethernet Link that provides gigabit communication between nodes, or between a node and a computer. This board is the main external interface of the system, and can communicate directly with any computer with 1000BASE-X network capabilities. The board is designed to work with Ethernet protocol 10/100/1000 and to use UDP/IP as the communication protocol.

Although the board is capable of connecting using the IEEE 802.3ab (1000BASE-X) [7] protocol, it cannot send pure user data at this speed due to the addition of several layers of protocol, maximum packet size limitations, and the particular hardware used. The board can, however, send pure user data up to 800 Mbps.

The NPCB currently is responsible for formatting the sonic array data, processing it and sending to a computer through the Ethernet link. Currently, the computer will further process the data using specially designed software [8] and apply the Independent Component Analysis algorithm in order to achieve source separation.

IV. Independent Component Analysis

Independent component analysis or ICA is a mathematical technique used for extracting hidden parameters that underlie in sets of random variables or signals.

ICA is a type of blind source separation method and common inputs sources are signals originated from audio, images or telecommunications [9].

This technique is based on the assumption that signals from different sources are statistically independent and statistically independent signals can be extracted from mixture signals. Therefore, the condition of source statistical independence must be fulfilled to the success of this technique.

ICA defines a model for the observed data that requires a database of samples in order to establish the necessary statistics. The model assumes that the data variables are linear combination of unknown variables, the unknown variables are assumed to be non-Gaussian and independent. The goal then becomes to find a transformation in which the components are as statistical independent as possible from each other.

This technique is related with methods such as principal component analysis [10] and factor analysis. The main distinction between ICA and these techniques is that while ICA founds a set of independent sources, principal component analysis and factor analysis finds sets of signals which are uncorrelated. This means that ICA recovers the original sources while the other two methods only find sets of the signals that can be locally uncorrelated but not necessary globally independent.

A typical example of deployment of the ICA technique is the problem of source separation. When there are mixtures of simultaneous speech signals that have been picked up by a microphones array it is desirable to have the original signals isolated. One way of doing this is through an algorithm implementing ICA. The algorithm accumulates statics of the income signals and then analyzes it, trying to isolate non-Gaussian behavior and independent characteristic that the signal can present. Such algorithm is computing intensive since it must accumulate and go through the signal samples performing complex operations. In theory the ICA algorithm can distinguish as many sources as independent variables are generated by the linear combination of the sources. This

Authorized licensed use limited to: Illinois Institute of Technology. Downloaded on October 02,2020 at 06:04:11 UTC from IEEE Xplore. Restrictions apply.

means on practice that we need at least the same number of sensors as sources to correctly separate the signals.

Efficient versions of the algorithm have being already deployed using different techniques such as the FastICA [5] that can be implemented efficiently in hardware platforms such as DSP processors and FPGA's. The ICA algorithm being used for this application is a based on the FastICA and is implemented in Matlab. Because of the nature of the ICA algorithm, this system can only be used to separate non-Gaussian and independent sources.

The algorithm implemented in this work goes beyond the ICA, it also implements pre and post processing for an efficient use of the ICA algorithm.

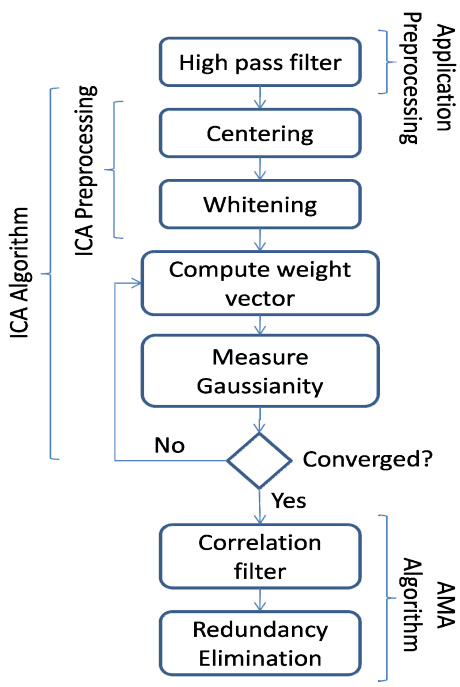

Fig. 6. Flowchart of the implemented algorithm.

Fig. 6 illustrates the full algorithm implemented for this system. The full algorithm encompasses a high pass filter, the ICA and the AIIA (AMA ICA Interpretation Algorithm) algorithm.

The first part of the algorithm is a second order high pass filter with corner frequency set to 100 Hz and it is meant to enhance the application of source separation by filtering out low frequency noise.After that, the signal enters the pre-processing stage of the ICA algorithm. On this stage the data is initially centered, that is simply the centering of the incoming variables, and consist in subtracting the mean vector for each variable making them a zero-mean variable. Still on the pre-processing stage the data is then whiten. The whitening process decorrelates the input signals as much as possible and equals their variance to one. This is required to get one step closer to the original signal, since the linear transformation added correlation between the original source signals.

The whitening process is a linear transformation where the covariance matrix of the input signals is equalized to the identity matrix. In this case it will be given by;

$$cov(X\ M) = I \tag{1}$$

where $\boldsymbol{X}$ is the new variable with the whitened data, $\boldsymbol{M}$ the variables containing the data and $\boldsymbol{I}$ the identity matrix. The transformation can be accomplished by several different methods; the most used being Principal Component Analysis [7].

After the ICA pre-processing the data enters the ICA algorithm itself. There are many different techniques of implementing the ICA algorithm, these techniques generally involve maximizing the non-Gaussianity of the variables feed to the algorithm. This is the basis of the ICA algorithms since the algorithm expects non-Gaussian sources for the signals.

The technique applied on this work for maximizing the non-Gaussianity of the input variables was negentropy. The concept of negentropy is based on entropy of the variables being processed. The more random a variable is, larger is its entropy and Gaussian variables have the largest entropy of any random variable with the same variance. Therefore negentropy can be used to measure the distance to normality that is a measure of non-Gaussianity. If a random variable is Gaussian its negentropy will be zero.

Mathematically negentropy is expressed by the difference between the differential entropy of the Gaussian random variable of the same covariance matrix as the random vector being processed and the differential entropy of the random vector itself,

$$J\,(Y) = H(Y_{gauss}) - H(Y) \tag{2}$$

where $\boldsymbol{Y}$ is the random variable vector and $\boldsymbol{Y_{gauss}}$ the Gaussian random variable vector with the same covariance matrix as $\boldsymbol{Y}$ and $\boldsymbol{H}$ the differential entropy. The differential entropy is defined as

$$(Y) = -\int f(Y)\log\big(f(Y)\big)\,dY \tag{3}$$

$\boldsymbol{f(Y)}$ is the density function of the random variable $\boldsymbol{Y}$.

The negentropy gives an excellent measurement of non-Gaussianity, however it is not proper to be implemented due to the complexity of the calculations involved, therefore a formula that gives an approximation of negentropy based on [5] was chosen, where

$$J(y)\bullet\left[E\{log\ (cosh(y))\} - E\left\{log\left(cosh(y_{gauss})\right)\right\}\right]^2 \quad . \tag{4}$$

The next step now is maximize the non-Gaussianity of the Y vector containing the mix signals, to do that vector X is created. Vector X is initially identical to Y and is recalculated by using the follow expression

$$X = W^T Y \tag{5}$$

where W is a weight vector that was initially chosen and X the new variable. The algorithm them proceed into an interactive process making use of

$$D_n = J(y_n) - J(y_{n-1}) \tag{6}$$

where D_n is the negentropy direction, $J(Y_n)$ the current negentropy and $J(y_{n-1})$ the last measured negentropy. If D_n is

Authorized licensed use limited to: Illinois Institute of Technology. Downloaded on October 02,2020 at 06:04:11 UTC from IEEE Xplore. Restrictions apply.

increasing it means that the non-Gaussianity is increasing on the other hand if D_n is decreasing the Gaussianity is increasing. Therefore the algorithm will interactively look for a new weigh vector W that will always increase the non-Gaussianity of X. The algorithm proceeds until it passes a threshold for D_n defined by the user that will mean that the algorithm converged. This new vector X contains then the estimated original signal.

The fastICA algorithm can provide redundant information and non optimal solutions. Most of the times ICA will provide redundant information when the sensor array feeds the algorithm more signals than real acoustic sources exist in the environment on wish the system is immersed in.

In principle the ICA algorithm will always generate the same number of outputs as the number of inputs. This is not always the case in this particular system because in order for the algorithm to converge fast, a limit of iterations was set, so if the limit is reached the ICA output is not generated, making it possible for the number of signals generated by the fastICA being equal to the number of real signals or even less. This limit of iterations on another hand increases a problem that is already presented on the algorithm that is the non optimal conversion.

In order to address the two above mentioned problems the AMA ICA Interpretation Algorithm (AIIA) was developed specifically to this system and is not guaranteed to work on other applications.

The AIIA algorithm is feed with the output vectors X_n from the ICA , it first try all combinations for correlation. This is achieved by applying the Pearson product-moment correlation coefficient also known as population correlation coefficient and given by

$$\rho_{X_n,X_m} = \frac{cov(X_n X_m)}{\sigma_{X_n}\sigma_{X_m}} \tag{7}$$

where ρ_{X_n,X_m} is the population correlation coefficient, X_n is the ICA variable that we want evaluate against a second ICA variable X_m, and σ_{X_n} and σ_{X_m} the respective standard deviation. The number of combinations that this algorithm will be calculating is given by $\binom{t}{2}$ where $\boldsymbol{t}$ is the number of ICA variables and $\boldsymbol{m}$ and $\boldsymbol{n}$ the index of the variable. These population coefficients are then compared with the population correlation coefficient threshold variable, set by the user, if the value is above the threshold, the signals are considered different, if bellow, the signals are considered the same. If the signals are considered the same then the second part of the algorithm is triggered and it will choose wish one of the same version of the signal will be chosen.

The algorithm then gets all the signals that where considered to be equals and then chooses those that have the smallest sample to sample mean derivative. This is done because it was observed on this system that signals that have multiple copies of themselves originated from the ICA algorithm will be very similar but will contain different levels of high frequency noise. This technique chooses the least noise of the signals.

It is also an integral part of the algorithm the power to identify variables that are mostly composed of noise our highly uncorrelated samples and can fail to be filtered out by the first part of the algorithm and therefore yield fake results. This is done by using the sample correlation coefficient derivate from the Person's correlated coefficient and given by

$$r = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{X_i-\bar{X}}{\sigma_X}\right)\left(\frac{Y_i-\bar{Y}}{\sigma_Y}\right) \tag{8}$$

where n is the sample number, X_i and Y_i the sample pair, σ_x and σ_y the standard deviation of the variables X and Y respectively. $\bar{X}$ and $\bar{Y}$ are the respective means of X and Y. The value of r is then compared with the sample correlation coefficient threshold set by the user, and then the signal will be deemed relevant or not, the user can always set the threshold to zero and bypass this step.

V. Results

First results of the system operating for its target application of source separation are now reported.

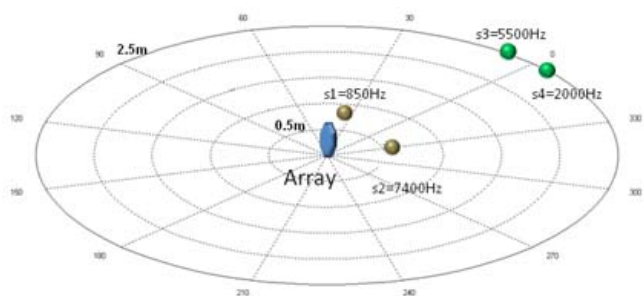

Fig. 7. Disposition of the loudspeakers relative to the sonic array. Microphones chosen for the test and the microphones used for this test.

The present test has as objective to probe the capability of the system in separating multiple sources and then evaluate wish signals coming out from the ICA algorithm are relevant.

The ICA algorithm can provide redundant information and non optimal solutions. Most of the times when the system feeds more signals to the algorithm than real signals coming from the environment in wish the system is immerse the system will provide redundant solutions

In order to validate the AIIA algorithm and at the same time test the system capabilities for multiple sources detection, a test was set where four sinusoidal acoustic waves are emitted by four different loudspeakers as presented on Fig. *8*.

Source s1 is set at a distance of 0.7 m from the array in a 45 degrees angle to the left of the array, s2 is set at a distance of 0.7 m and 45 degrees to the right of the array, s3 is set a distance of 2.5m and 5 degrees left of the array and s4 at a distance of 2.5 m and 5 degrees to the right of the array. The frequency of the sources s1,s2,s3 and s4 are respectively 850 Hz,7400 Hz,5500 Hz and 2000 Hz.

Authorized licensed use limited to: Illinois Institute of Technology. Downloaded on October 02,2020 at 06:04:11 UTC from IEEE Xplore. Restrictions apply.

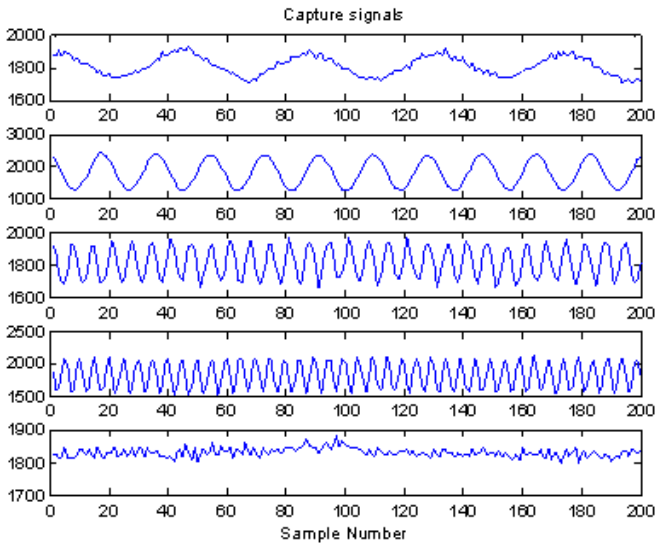

Fig. 8 The five individually captured signals are from top to bottom s1,s4,s3,s2 and the background noise.

The first step of this experiment was to acquire the data independently for each source, one source at the time was stimulated and acquired. After that, the array acquired data with no source so to have measurements of the background noise, these measurements can be observed on Fig. *9*. After the individual signals where acquired for posterior comparison, the system was activate to collect data of all sources at the same time and the result of this data acquisition is provided by Fig. *9*.

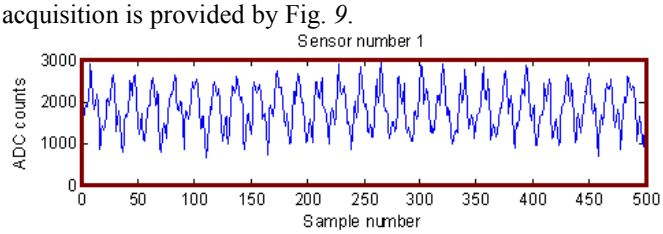

Fig. 9 On the top, a snapshoot of the microphone channel one array acquiring the mixed data.

After the data was acquired it was immediately sent for the ICA algorithm, was run in with four microphones. It's possible to observe in these results that the ICA algorithm was not able to distinguish the four original signals even with the same number of sources available. The AIIA algorithm performed as expected and eliminated two similar vectors.

Although arguable that due to background noise the system still did not have the same number of acoustic sources as sensors, the system found only two signals. That shows that noise and system non-linearity affect the algorithm strongly. The same test was emulated in Matlab with perfect sine waves combined by a linear transformation feed to the algorithm and no noise. The algorithm was able then to found all original signals.

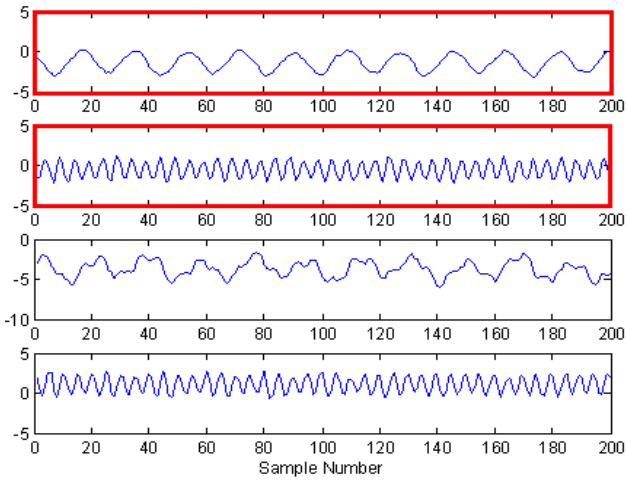

Fig. 10 Four microphone result. The two top results were selected by the AIIA algorithm.

The next test was performed with twelve sensors and it was a challenging test both for the ICA algorithm as for the AIIA. Both algorithms performed very well on this configuration and the results obtained by the ICA are displayed on Fig. *11* and the AIIA selections displayed by Fig. *12*.

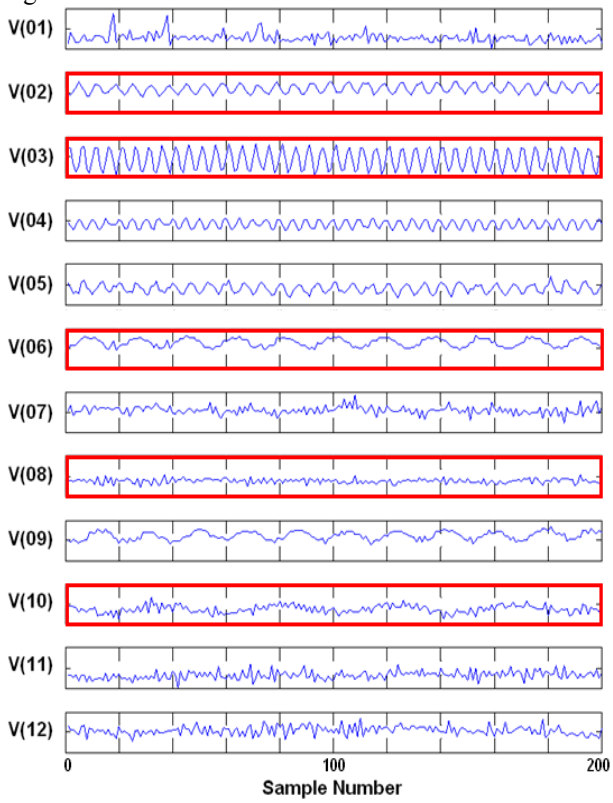

Fig. 11 Twelve vectors generated by the ICA algorithm. Signals in red are the vectors chosen by the AIIA.

This test is the one that provided the highest level of correlation with the original signals. After this test, the

Authorized licensed use limited to: Illinois Institute of Technology. Downloaded on October 02,2020 at 06:04:11 UTC from IEEE Xplore. Restrictions apply.

algorithm was run with 24 and 52 sensors and the results were very poor due to non convergence of the ICA algorithm. In order to the algorithm converge the limit number of interactions for the ICA algorithm needed to be increased exponentially what caused the time o processing to increase a lot. Even after that, the results were not better than the ones reached with configuration the twelve sensors configuration.

The number of sources was also decreased to three and two and the relation observed was that the number of microphones used to produce the best results in this system is approximately three times the number of real sources, adding mores microphones does not depreciate the results as long as the algorithm converges.

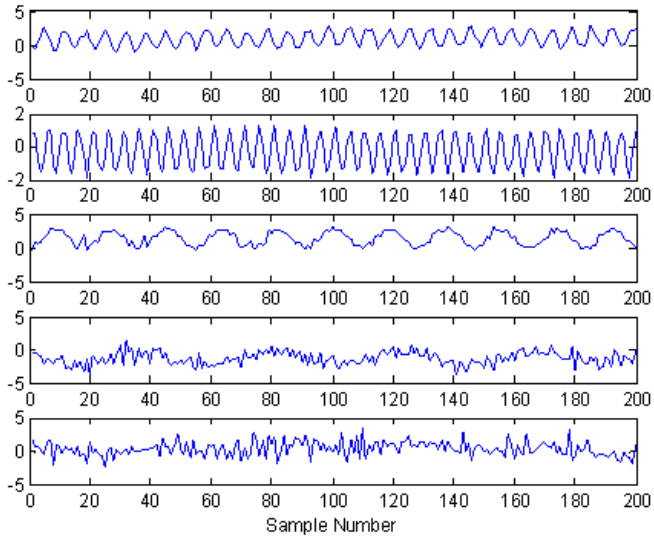

Fig. 12 The five vectors chosen by the AIIA algorithm when twelve microphones were utilized.

The AIIA results proof that the algorithm is pretty reliable for this system as long as the population correlation coefficient threshold and the sample correlation coefficient threshold are calibrated beforehand. The calibration of these coefficients is strait forward and just uses a sine wave and the local background noise, usually the default values will work without the need of change as long as the gain calibration of the array does not change.

VI. Conclusion

This work demonstrates the implementation of a system based on Independent Component Analysis dedicated to solve the problem of blind source separation in real-time. Results of the array acquiring data and performing sound separation were provided and studied. The results show that the array is capable to perform its designed task when integrated with the ICA algorithm.

It is also part of this work the description and utilization of the CAPTAN architecture, providing its characteristics, capabilities and limitations. Furthermore its application using the specially designed AMA sonic array is evaluated. The AMA array was integrated with the ICA, AIIA and sound localization algorithms and its capabilities demonstrated.

This work demonstrates that it is possible to integrate key technologies such as Independent Component Analysis, MEMS, high performance FPGA, and Gigabit Ethernet to produce a very compact network based blind source separation system with high performance.

VII. References

[1] Turqueti, M., Rivera, R., Prosser, A., Andresen, J. and Chramowicz, J., "CAPTAN: A hardware architecture for integrated data acquisition, control and analysis for detector development", IEEE Nuclear Science Symposium, 3546-52, October, 2008.

[2] Weinstein E., Steele, K., Agarwal, A., and Glass, J., "A 1020-Node Modular Microphone Array and Beamformer for Intelligent Computing Spaces", MIT/LCS Technical Memo, MIT-LCS-TM-642, April, 2004.

[3] Knowles acoustics, SPM204 and SPM208, Datasheet, Rev C, 2006.

[4] Benesty J., Chen, J., Huang, Y., "Microphone Array Signal Processing", Springer Topics in Signal Processing, first edition , Springer, Vol. 1,2008.

[5] A. Hyvärinen, J. Karhunen, E. Oj, *Independent Component Analysis*, Wiley, New York, 2001.

[6] Xilinx, Inc.,Virtex-4 FPGA User Guide, UG070 (v2.6) December 1, 2008.

[7] Jolliffe,I.T, *Principal Component Analysis*, Springer-Verlag, New York, 2002.

[9] Stone J.V., *Independent Component Analysis: A Tutorial Introduction* Volume 1, MIT Press, Cambridge, Mass., 2004.

[10] Nordholm, S., Siow, Y. L., "Speech Signal Extraction Utilizing PCA-ICA Algorithm With a Non-Uniform Spacing Microphone Array," Proceeding of the IEEE International Conference on Acoustics, Speech and Signal Processing , Vol. 5, pp. V-V, May, 2006.

Authorized licensed use limited to: Illinois Institute of Technology. Downloaded on October 02,2020 at 06:04:11 UTC from IEEE Xplore. Restrictions apply.