Arduino Due Implementation of an Algorithm for Blind Source Separation using Matlab Simulink

M. Mekhfioui, R. Elgouri,A. Satif, L. Hlou

***Abstract:** This work describes the implementation of an algorithm of the blind source separation in the Adruino Due card as free and low cost material. The algorithm must be developed to adapt the inputs and outputs of the Arduino board to the actual installation. Using reference signals generated by two generators (GBF), this work has proved that the algorithm Second Order Blind Identification (SOBI) is the most efficient to analyze our signal database, so the separation is performed on the Arduino Due using the SOBI algorithm to validate its performance in real time.*

***Keywords :**Blind Source Separation, BSS, SOBI,ACI, Arduino Due, Matlab/Simulink, Real Time Workshop.*

I. INTRODUCTION

Most often in our environment, several signal sources emit in this environment, signals propagate towards a sensor, undergo a transformation by the propagation medium and combine to compose mixtures that are sometimes complex. So how can we then extract the source signals that are at the origin of these mixtures?.

The Blind source separation proposes a solution to this problem and consists in estimating a group of unknown source signals from their observed mixtures when there is very little information available on the mixing model,without any additional assumptions. This source separation technique has been used in different scientific and technological domains, like telecommunications, biomedical signal processing. Acoustics and imaging.

The first works were published by HeraultJutten and Ans in 1980 [1] about mixtures called instant linear.

Revised Manuscript Received on December 30, 2019.
***** Correspondence Author

M.Mekhfioui*, Laboratory of Electrical Engineering and Energy Systems (LEEES), FSK, Ibn Tofail University, Kenitra, Morocco. Email: mohcin.mekhfioui@uit.ac.ma

R.Elgouri,, Laboratory of Electrical Engineering and Telecommunications Systems (LEETS), ENSA Ibn Tofail University, Kenitra, Morocco. Email: rachid.elgouri@uit.ac.ma

A.Satif, Laboratory of Electrical Engineering and Telecommunications Systems (LEETS), ENSA Ibn Tofail University, Kenitra, Morocco. Email: amal.satif@uit.ac.ma

L.Hlou, Laboratory of Electrical Engineering and Energy Systems (LEEES), FSK, Ibn Tofail University, Kenitra, Morocco. Email: laamari.hlou@uit.ac.ma

Other works, represent a biological problem that initiated studies on source separation, and consists in treating the responses emitted by muscles subjected to different types of excitations [2].

C. Jutten and J. Herault have presented A first mathematical formalism of the BSS problem and an algorithm to reach a solution in 1991 [3].Then the Comon works in 1994 [4] allowed the mathematical formalization in the simplest case of an instantaneous linear mixture leading to the concept of Independent Component Analysis (ACI).

Previously, few studies are carried out on the real-time implementation of blind source separation. Zhang and Zhu [5] carried out an implementation of the Fast ICA algorithm in a DSP card for blind source separation furthermore Tornez-Xavier and all worked on FPGA Implementation of the ACI Algorithm.

Meyer-Baese and All [6] investigated the capacity and performance of FPGA independent component analysis ICA algorithms to realize real-time separation of blind sources, as well as, Singh and all [7] proposed an FPGA implementation of Blind Source Separation (BSS) using a new ICA algorithm based on the rotation of the Jacobian matrix and the Matlab system generator blocks. In 2018.

On the other side, none of these works has tackled the separation of sources in real time using the Arduino card as free and low cost material.

In this work, we present a comparison with the different algorithms of the BSS based on the calculation of the SIR (signal-to-interference ratio) and the PI (separabilityPerformance Index), the best algorithm obtained will be implemented in the Arduino board for tested the performance of the latter in real time.

This work is structured as follows: Section II presents the principle of the blind source separation BSS and its mathematical model, the Arduino Due card and the signals used, Section III presents the the module design and the results of the simulation, finally the Section IV concludes this document.

II. MATERIAL AND METHODOLOGY

A. About Blind Source Separation

The Blind Source Separation (BSS) is based on estimating unknown source signals from their observed mixtures when there is very little information available on the mixing model [8].

The principle of the BSS can be depictedin Fig.1, It is to find in the ideal case the matrix W of size $n \times m$ which provides the output vector:

$$\boldsymbol{y}(\boldsymbol{k}) = \boldsymbol{W}\,\boldsymbol{x}(\boldsymbol{k}) = \boldsymbol{W}\,\boldsymbol{H}\,\boldsymbol{s}(\boldsymbol{k}) \approx \boldsymbol{s}(\boldsymbol{k}) \quad (1)$$

The estimated sources are obtained from the vector $s(k)$ and their related projections for the various sensors are determined from the estimated mixture array: $H=W^{-1}$.

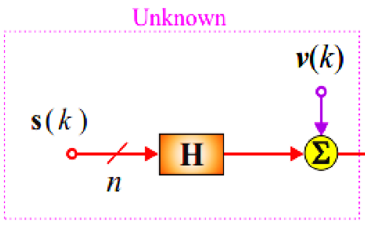


Fig. 1: Principle of the separation of sources [8]

The efficiency of the algorithms of the Blind Source Separation BSSis evaluated by calculation the separability performance index (PI), is obtained by the equation below.

$$PI = \frac{1}{n(n-1)} \sum_{i=1}^{n} \left\{ \left(\sum_{k=1}^{n} \frac{|g_{ik}|}{max_j |g_{ij}|} - 1 \right) + \left(\sum_{k=1}^{n} \frac{|g_{ki}|}{max_j |g_{ji}|} - 1 \right) \right\} (2)$$

Where g_{ij} is the (i, j) item of the global matrix of the system $G = WH$and$max_i(g_{ij})$presents the maximum value from the itemsof the i^{th} line vector of G. Similarly the term $max_j(g_{ji})$ corresponds to the maximum value from the itemsofthe j^{th}spine vector of G.

When the optimal separation is achieved, the separability performance index is zero. In reality, a PI value of about 10^{-2} denotes a relatively high performance in terms of separation [8].

B. Arduino Due card

The Arduino Due is a programmable card based on a low costARM Cortex-M3 microcontroller. It contains 54 digital pins (including 12 PWM), an 84 MHz clock, 4 UARTs, 12 analog inputs, a JTAG header Fig.2. [9]

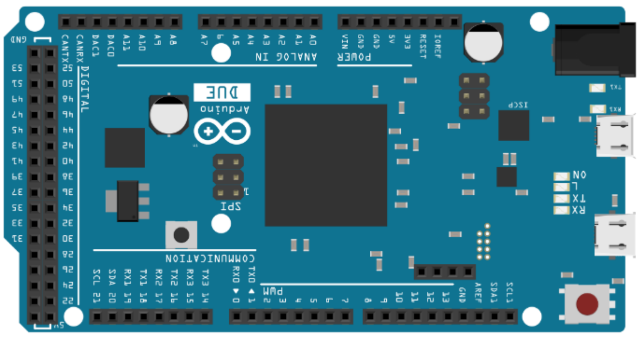


Fig. 2: Schematic diagram of an Arduino Due card

The analog input ports of this Arduino have a resolution of 10 bits (values in the range 0 to 1023), implying that an input voltage of 0V is as value 0, and an input voltage of 3.3V is represented as a value 1023. On the other hand, the DAC analog outputs have a resolution of 8 bits, with the option of operating with a 10bit resolution the value of 0 creates an output potential of 0.55v and the value 1023 is transformed into an output of 2.77v Table 1.

Table1: Technical specification of the Arduino Due

Microcontroller	AT91SAM3X8E
Clock Speed	84 MHz
SRAM	96 KB
Flash Memory	512 KB
Digital I/O Pins	54
Analog Output Pins	2
Analog Input Pins	12
Operating Voltage	3.3V
Input Voltage (limits)	6-16V
Input Voltage (recommended)	7-12V
DC Current for 5V Pin	800 mA
DC Current for 3.3V Pin	800 mA
DC Output Current	130 mA

III. RESULTSANDDISCUSSION

A. Software Simulation

The basic signal used in our experiment consists of two signals:

✓ Square signal: This discontinuous signal represents the quantities of very fast variations (transmission of pulses, shock and friction of parts, ...).

✓ Triangular signal: This signal describes the linear variations (magnetic excitation, speed of rotation, temperature, ...).

In order to simulate the propagation environment, the signals will be mixed linearly by a mixing matrix in the Arduino programming board $A = \begin{pmatrix} 0.12 & 0.65 \\ 0.76 & 0.25 \end{pmatrix}$to produce the input signals x_1 and x_2.

In our study we have used the ICALAB[8] grouping several algorithms of the Blind Source Separation BSS that we tested (20 algorithms tested) to determine the algorithm that gives the higher performance. However, tests have shown that only 9 algorithms are stable:

- ✓ SYM-WHITE: [14]
- ✓ SOBI:[10];
- ✓ SAD:[11];
- ✓ JADE:[13];
- ✓ AMUSE : [15];
- ✓ POWERICA: ICA [16];
- ✓ FPICA:[17];
- ✓ FOBI:[12];
- ✓ UNICA: ICA [18];

Table 1. The performance index obtained for the algorithms BSS tested

Algorithm	PI	SIR
SOBI	0,001	81,40
JADE	0,001	70,23
AMUSE	0,002	62,56
FOBI	0,005	58,28
POWERICA	0,006	56,75
SYM-WHITE	0,007	54,65
FPICA	0,009	54,87
SAD	0,023	49,47
UNICA	0,024	41,25

Results show that the SOBI algorithm (Second Order Blind Identification) exhibitsa more appropriate separation technique in the present case.

The Fig. 3 shows the origin signals (s_1 and s_2), the multiplexed signals (x_1 and x_2) and the output signals (y_1 and y_2) for the SOBI algorithm.

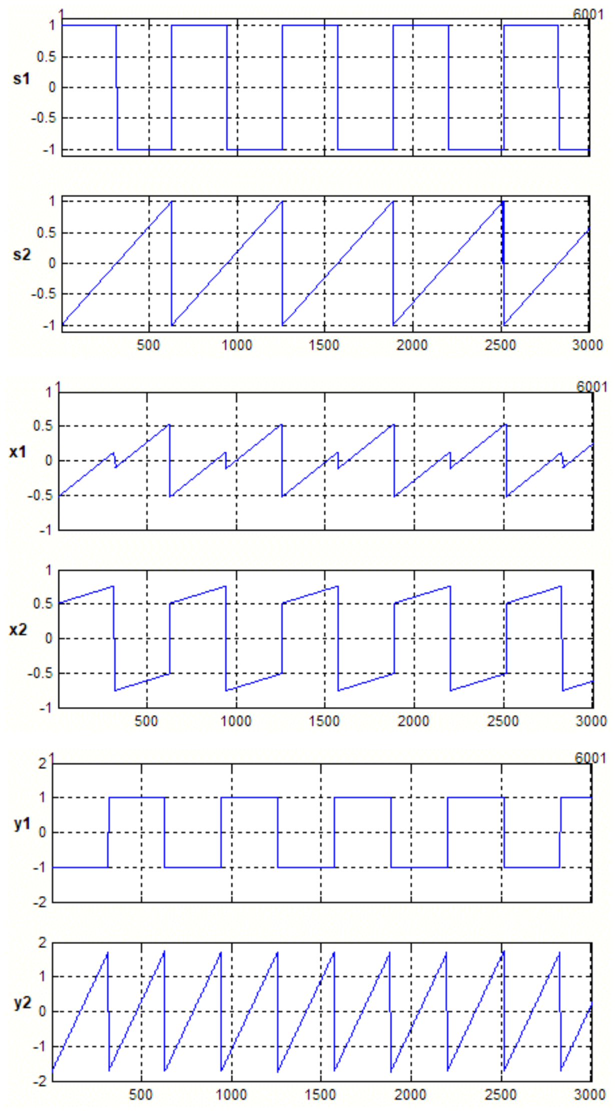


Fig.3: Simulation results of the SOBI algorithm

B. Implementation and Experimental Results

In this part, the SOBI algorithm was implementedin an Arduino Due board. To test the implementation,two generators GBF was used to produce the input signals x_1 and x_2, and an oscilloscope to view the output signals Fig.4.

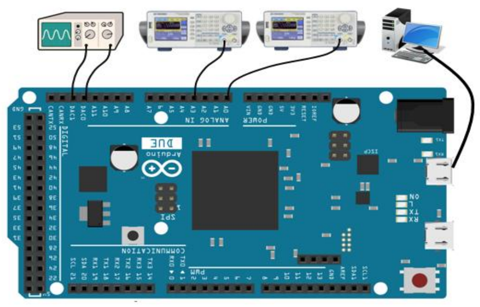


Fig.4: Hardware setup for system separation

To use arduino block in matlab, Simulink Support Package Library for Arduino hardware must be installed,itcan be obtained and installed by clicking on the Add-On on Matlab. When installation is complete, the library can be added to the Simulink Library Browser as a Simulink support package for Arduino.

Simulink needs to be configured and operated with Arduino Due. In the simulation options, the external mode is enabled to run in real time with the external hardware. When execution is requested, RTW converts the Simulink block diagram into C/C++ code, in order to compile and download the code into the microcontroller. The code is run in real time with the chosen sampling time in the blocks and the selected signals are sent to Simulink via USB interface. The figure 5 show the diagram of the Real Time Workshop [19].

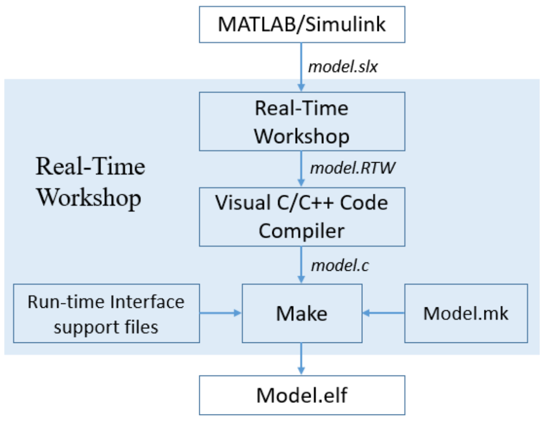


Fig. 5: Flow diagram of the procedures of implementation the model on Arduino Due

The Fig. 6 shows a design of blind source separation by the real-time SOBI algorithm using the Arduino card under Matlab / Simulink.

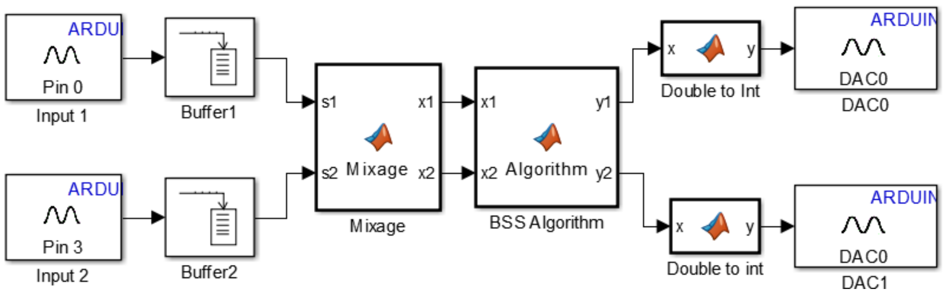
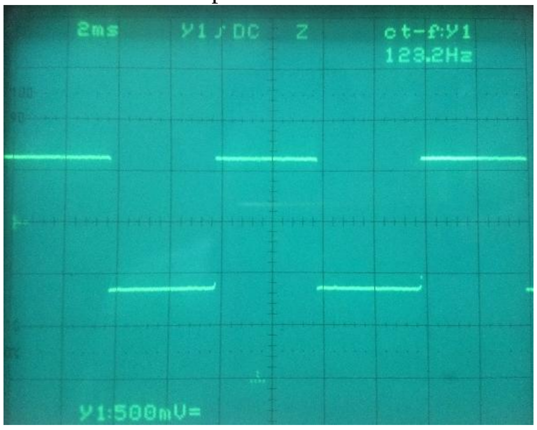


Fig. 6: Model Real-time Separation of tow audio signal

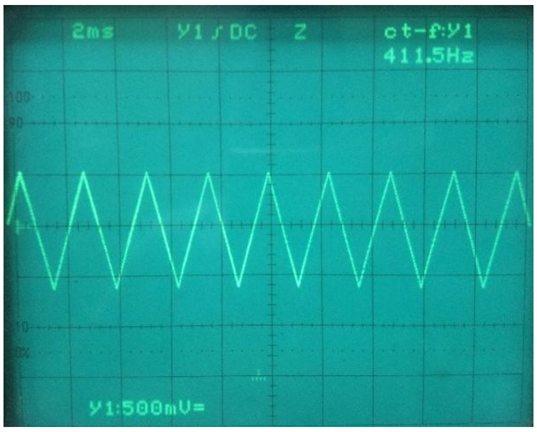
In addition, we designed a reliable module-based S function to evaluate the performance of the Blind Source Separation technique by calculating the Signal to Interference (SIR) and separability Performance Index (PI) values.

In real application, the SOBI algorithm has been implemented and tested with two generators and an oscilloscope, if the number of outputs is greater than two, we used the switches on the Arduino board to choose the output to be shown on the oscilloscope.

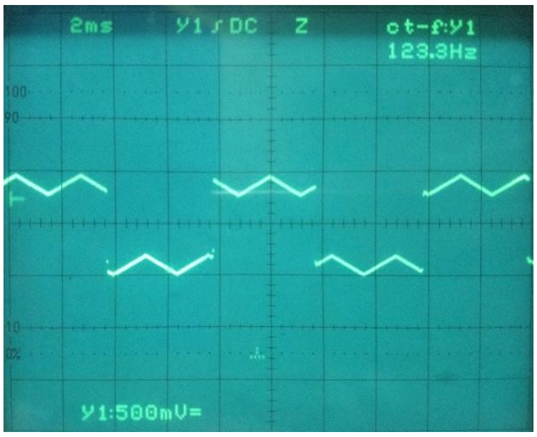
Using our workstation the input signals in real time using the SOBI algorithm and Arduino Due card could be seperated. To observe the different signals (source signals, mixing signals and estimated signals) we should change the binary value of the switches of the card (Fig. 7).



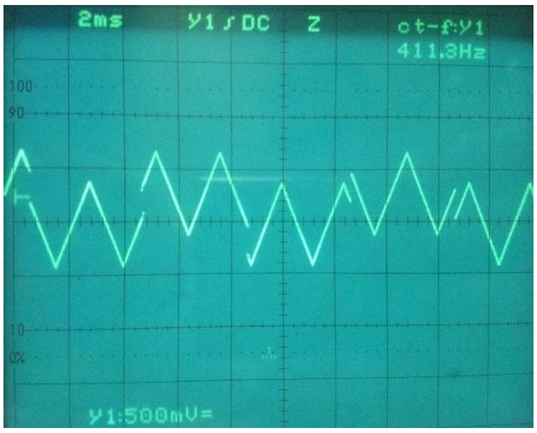
Source signal s1



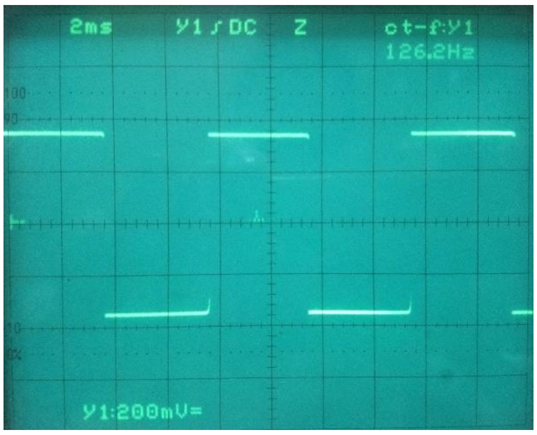
Source signal s2



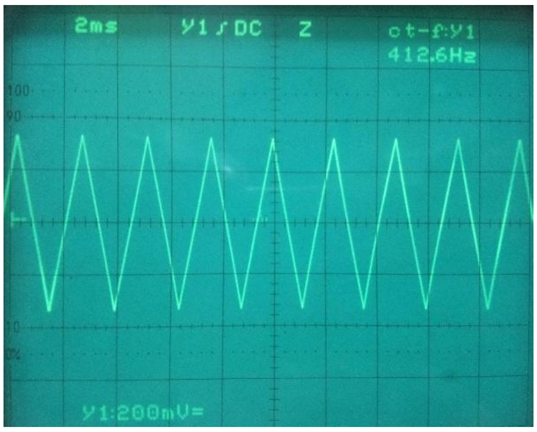
Mixing signal x1



Mixing signal x2



Estimated signal y1



Estimated signal y2

Fig. 7: Result of Real time implementation using the SOBI algorithm for an audio signal

The results of the implementation show a similar behavior as the results obtained by simulation on Simulink, which confirm the accuracy of the method.

C. Discussion

We used the Arduino Due to separate two linearly mixed signals by an A matrix, using the best-performance algorithm obtained by calculating the Performance Index of separability. This study aims to reproduce the performance of the implementation of the Second Order Blind Identification (SOBI) algorithm as expected in Matlab.

The experiment carried out in this paper is still in the onset and provide a background to be developed in future works concerning the performance of the SOBI algorithm in real time, which is a promising area of research.

IV. CONCLUSION

In this work, we presented a comparison of the blind separation of sources using different algorithms of the ICALAB tool on two signals picked up by two generators and multiplied by a linear matrix A, based on the calculation of the Performance Index of separability (PI) and Signal to Interference (SIR), it is found that the SOBI algorithm (Second Order Blind Identification)is most efficient to analyze database. To validate this performance in practice and in real time, we implemented this algorithm in the Arduino Due board as free materials and low cost, using two generators and an oscilloscope, we obtained accurate results similar to the results of Matlab.

REFERENCES

1. C. Jutten and J. Herault, "Space or time adaptive signal processing by neural network models," AIP Conference Proceedings, pp. 206-211, 1986.
2. C. Jutten and A. Taleb, Source Separation: From Dusk Till Dawn, 2000.
3. C. Jutten and J. Herault, «Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture,» Signal Processing, vol. 24, n° %11, pp. 1-10, 1991.
4. P. Comon, «Independent component analysis, A new concept?,» Signal Processing, vol. 36, n° %13, pp. 287-314, 1994.
5. M. Zhang, M. Zhu and W. Ma, «Implementation of FastICA on DSP for Blind Source Separation,» Procedia Engineering, vol. 29, pp. 4228-4233, 2012.
6. U. Meyer-Baese, C. Odom, G. Botella and A. Meyer-Baese, «Independent component analysis algorithm FPGA design to perform real-time blind source separation,» Independent Component Analyses, Compressive Sampling, Large Data Analyses (LDA), Neural Networks, Biosystems, and Nanoengineering XIII, vol. 9496, 2015.
7. V. Singh, V. Somani et J. Manikandan, «FPGA implementation of blind source separation using a novel ICA algorithm,» IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), pp. 67-71, 2017.
8. H. Arahmane, E. Hamzaoui and R. Cherkaoui El Moursli, «Neutron Flux Monitoring Based on Blind Source Separation Algorithms in Moroccan TRIGA MARK II Reactor,» Science and Technology of Nuclear Installations, vol. 2017, 2017.
9. T. Xiaogang, W. Sun'an and L. Jiong, «Blind Source Separation for Complex-Valued Signals Using Generalized Autocorrelation,» Mathematical Problems in Engineering, vol. 2018, n° %18076468, p. 9, 2018.
10. «Arduino Due,» Arduino Inc, [En ligne]. Available: https://store.arduino.cc/due. [Accès le 15 November 2019].
11. C. Wang, J. Wang et T. Zhang, «Operational modal analysis for slow linear time-varying structures based on moving window second order blind identification,» Signal Processing, Vols. %1 sur %2169-186, p. 133, 2017.
12. X. Li and X. Zhang, «Sequential Blind Extraction Adopting Second-Order Statistics,» IEEE Signal Processing Letters, vol. 14, n° %11, pp. 58-61, 2007.
13. K. Nordhausen et J. Virta, «An overview of properties and extensions of FOBI,» Knowledge-Based Systems, vol. 173, pp. 113-116, 2019.
14. J. F. Cardoso et A. Souloumiac, «Blind beamforming for non-Gaussian signals,» IEEE Proceedings F - Radar and Signal Processing, vol. 140, n° %16, pp. 362-370, 1993.
15. A. Cichocki, S. Osowski and K. Siwek, «Prewhitening Algorithms of Signals in the Presence of White Noise,» chez 6th International Workshop Computational Problems of Electrical Engineering, Zakopane, Poland, 2004.
16. L. Tang, R. Liu, V. C. Soon et Y. Huang, «Indeterminacy and identiability of blind identication,» IEEE Transactions on Circuits and Systems, vol. 38, n° %15, pp. 499-509, 1991.
17. S. Ding, «A Power Iteration Algorithm for ICA Based on Diagonalizations of Non-Linearized Covariance Matrix,» chez First International Conference on Innovative Computing, Information and Control, Beijing, 2006, pp. 730-733.
18. A. Hyvärinen et E. Oja, «A Fast Fixed-Point Algorithm for Independent Component Analysis,» Neural Computation, vol. 9, n° %17, pp. 1483-1492, 1997.
19. S. Cruces, A. Cichocki and L. Castedo, «Blind source extraction in gaussian noise,» chez 2nd International Workshop on Independent Component Analysis and Blind Signal Separation (ICA'2000), Helsinki, Finland, 2000, pp. 63-68.
20. «Arduino Support from MATLAB,» MathWorks, Inc, [En ligne]. Available: https://www.mathworks.com/hardware-support/arduino-matlab.html. [Accès le 13 November 2019].