A complete simulation tool for sound control applications over wireless acoustic sensor networks

Christian Antoñanzas, Juan Estreder, Gema Piñero, Miguel Ferrer, María de Diego, Alberto González

Audio and Signal Processing Group
Inst. of Telecommunications and Multimedia Applications (iTEAM)
Universitat Politècnica de València
8G Building - access D - Camino de Vera s/n - 46022 Valencia (Spain)
Corresponding author: chanma@iteam.upv.es

Abstract

The Wireless Acoustic Sensor Networks (WASN) are considered as a cheap, flexible and efficient solution for different acoustic applications, but their performance can be difficult to analyse due to the complex interaction of sounds through the air. Moreover, impairments of the wireless network, acoustic transducers, data converters, and the delays of the operating systems can greatly affect the acoustic performance of a particular sound control application. In this paper, we explain the main issues of a developed Matlab tool to model and simulate sound control systems over WASNs. The tool is able to simulate different network deployments and communication procedures, different acoustic systems and different algorithms for sound control. As a complement to the simulation tool, we also explain how to deploy a real WASN by means of mobile devices with Android operating system. We also indicate basic Android instructions to manage the Wi-Fi connection between mobile devices (smartphones and tablets) and the Bluetooth connection between devices and wireless speakers. Finally, current technological limitations affecting WASN performance are also discussed.

Keywords: Wireless acoustic sensor networks, sound control, matlab, android, bluetooth, wi-fi.

1. Introduction

The Wireless Sensor Networks (WSN) are considered as a cheap, flexible and efficient solution for environmental and habitat monitoring, as well as for monitoring and maintenance of industrial equipment [1, 2, 3] among other tasks. They are in the scope of research since the beginning of this century, although their commercial used is not as spread as it was expected. From the very first moment, different acoustic applications were proposed [4, 5] paving the way for the wireless acoustic sensor networks (WASN) whose particular characteristic with respect to the WSNs is that they use microphones, or any other kind of acoustic sensor. In a WASN, the node consists on one or more microphones connected to a processor with some kind of communication and computation capability. Applications that make use of these kind of acoustic nodes are numerous, see [6, 7] and references therein. They focus on the estimation of a common signal or parameter that can be measured by all the nodes [7], or on the estimation of node-specific signals sharing some common properties or parameters [8]. Another typical feature of this kind of node regards on its processing unit, which is usually dedicated to sound recording, control and transmission, although it can eventually perform some signal processing algorithms before the transmission.

However, if the WASN has to be used for applications that control the sound field, the node should be able to emit sounds through a loudspeaker or actuator. Moreover, the network should focus not only on the estimation of a particular signal or some related parameter, but also on the design of the signals that will feed the loudspeakers and will control the emitted sound [9]. In this sense, throughout this paper we will consider a generic acoustic node that can record and process signals, communicate to other nodes to exchange local and network information, and emit sounds to modify its own environment.

From a practical point of view, a WASN can be implemented using smartphones or tablets as acoustic nodes. Indeed, these devices have one or two microphones, one loudspeaker, and their processors exhibit a similar performance to those of notebooks, even better in some recent models. They can also communicate to other devices

An acoustic node is a device capable of obtaining information from one or more sensors (microphones) and generating signals via one or more actuators (loudspeakers) to control the sound field.

via Wi-Fi IEEE 802.11 or Bluetooth connections. In this sense, the number of sound processing applications on mobile devices has been increasing during the last years. Most of them are intended for entertaining, but not only. Some recent examples are [10], where several mobile devices reproduce the same sound simultaneously in order to provide an immersive sound experience as if they were inside a theatre, and [11], where the performance of a wide range of applications used to measure the sound pressure level is thoroughly examined.

The remainder of the paper is as follows. In Section 2, a complete simulation tool in Matlab is presented. This tool can simulate different network deployments and different sound control applications over these networks, as well as evaluate some of their main performance parameters. In Section 3, the description of the basic parameters in ***Android*** for the deployment of a real WASN using mobile devices and wireless loudspeakers is given. Finally the main conclusions are summarized in Section 4.

2. Simulator of sound field controller over WASNs

As stated in Section 1, we define an acoustic node as a device capable of obtaining information from one or more sensors (microphones) and generating signals via one or more actuators (loudspeakers) to control the sound field. It can also communicate with other network nodes. Based on this, a tool to model and simulate sound control systems and their behaviours is proposed in the following, but first of all it is important to explain the three different parts regarding WASNs (see Figure 1). Although they may share information, these three parts should work independently one of each other:

1) *Algorithms*: they define the processes to be performed on signals and the interaction between the system and the input-output devices (microphone-loudspeakers) as well as with other network elements.

2) *Communication system:* it allows information exchanges among the nodes and can simulate real effects of communications as latencies, transmission errors, information loss, etc.

3) *Acoustic system:* it models the transformation suffered by the signals from their reproduction at the loudspeakers till their recording at the microphones. It simulates the physical propagation and reflection of the sound by means of impulse responses measured at different points of the closed space.

Another important feature of the simulation tool is that each node can use distributed signal processing if the communication network allows to it, which means that an output signal is obtained at each node as a result of processing both the local input signal and the information received from other nodes. Although each node processes the signals independently, each one is relevant to the working of the global system.

The simulator presents other benefits like the storage, reproduction and analysis of all the audio signals or parameters in the simulation as well as tools for the definition, modification or study of the acoustic systems. The tool simulation has been developed to analyse the algorithms that define the sound field control applications in a real system without the need to program them on a DSP plat-

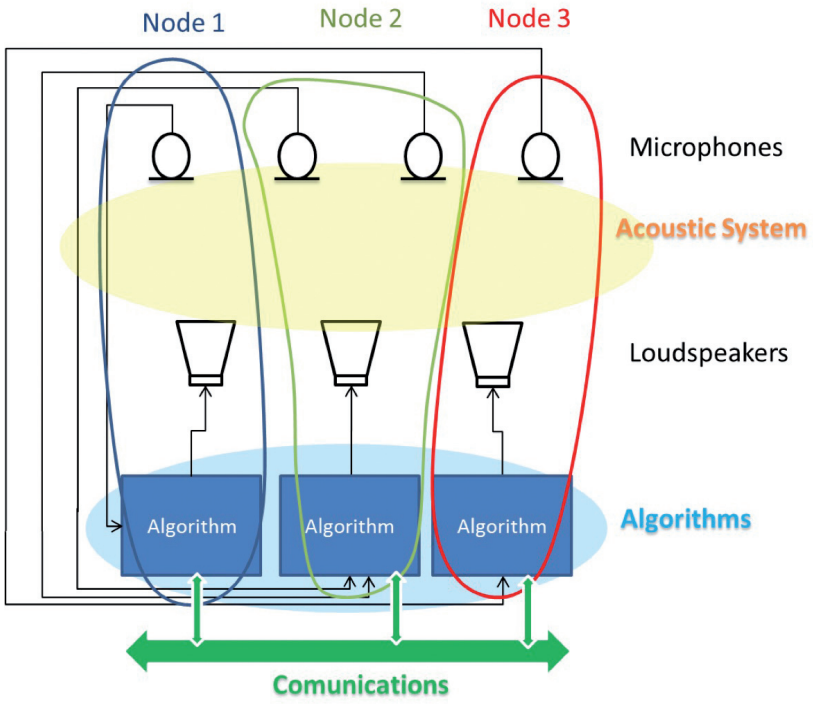

Figure 1. Diagram of a sound control system with multi-channel nodes.

form or to have an acoustic system in situ. Several types of communication errors between the network and the nodes and even model the data acquisition impairments in real-time applications can be simulated. In conclusion, this tool is intended to fill the gap between the initial mathematical idea of an algorithm and its final programming on a digital platform.

2.1. Definition of the data type "struct"

The simulator is based on the MATLAB data type called "struct", which is a structured array of data sets organized by groups called fields. Each field can contain any type of data of any size. Structures can be built in two ways:

- Using the "struct" function **(structArray = struct ('field1', val1, 'field2', val2, ...))** where the arguments are field names and their corresponding values.
- Using an assignment statement that assigns data to individual fields **(structArray.field1='val1'; structArray.field2=val2; ...)**.

Some of the most important features of this data type are:

- All structures in the array must have the same number of fields.
- All fields must have the same field names.
- Fields of the same name in different "struct" variables can contain different types or sizes of data.
- Besides, a "struct" provides hierarchical storage mechanisms to contain different types of data.
- Nested structures are allowed and vectors and matrices of structures can be created.
- It is possible to add a new field to a structure at any time.
- Moreover, the structures have optimized functions for specific operations. See MATLAB documentation online for extensive information [12].

2.2. Description of the simulation tool

In this section, a brief description of the different steps to perform the simulation of an acoustic network model is defined. We define acoustic network model as the model based on the configuration of all the parameters needed to simulate a sound field control application.

The first step is to initialize the acoustic network through the configuration of all the system processes:

- Acoustic system configuration: This setting loads the file containing the acoustic channel impulse response between each loudspeaker and each microphone located in the enclosure. In this part, the number of loudspeakers and microphones that will manage each node is defined. It is possible to create as much multichannel nodes as number of loudspeakers and microphones.
- Algorithms configuration: The algorithm to be executed by each node is selected depending on the application to simulate. Besides, the different parameters needed for the operation of each algorithm are defined in such a way that several nodes can share the same algorithm with different configuration parameters.
- Communication system configuration: It defines the type of topology of the communication network and how the nodes are collaborating. For example, a node can interchange information with a small group of nodes or only with its neighbour nodes. On the other hand, the communication network can introduce latencies in the transmission of information to the nodes hence it is possible to simulate this delay in number of samples at each node.
- Data acquisition system configuration: The tool tries to simulate the data acquisition that would be carried out in real-time applications over asynchronous networks. Each node has its own clock and stores, process and sends data independently from the rest. Similar to a DSP performance, a node is in a permanent state of low consumption until an interruption occurs due to the arrival of data. At that time, the node performs the corresponding processing. Such processing ends with another interruption returning to the initial state of low consumption of the node. Thus, we define two interrupt flags: reading/writing (R/W) and processing time. R/W interruptions indicate the moment at which the node reads the new data and writes the previous data. Processing time indicates the time it takes to process a sample block. If the node can read new data (R/W interruption is active) but the node is still processing previous data (processing time interruption is active too), the new data will be lost. It also defines the sampling time as the minimum unit of time at which the simulator works. So, the acoustic system works sample-by-sample in the time domain (to approximate us as close as possible to the continuous domain) but the nodes can execute algorithms implemented with blocks of samples in the frequency domain too. Similarly, the communication network works sample-by-sample independently of the other processes. Summarizing, at the same time and regardless of the system unit of time, each node can work with a different size block, a different sampling frequency and a different processing time.

Once considered the configuration data explained above, the network structure is created and the acoustic network model is characterized. Notice that the network structure addresses the information but does not process it. Therefore, the structure must have particular fields containing the variables that will be used within the algorithms, the acoustic system and the communication system for data processing.

Some of these fields are:

- State: Field containing the input and output variables of each process in the system.

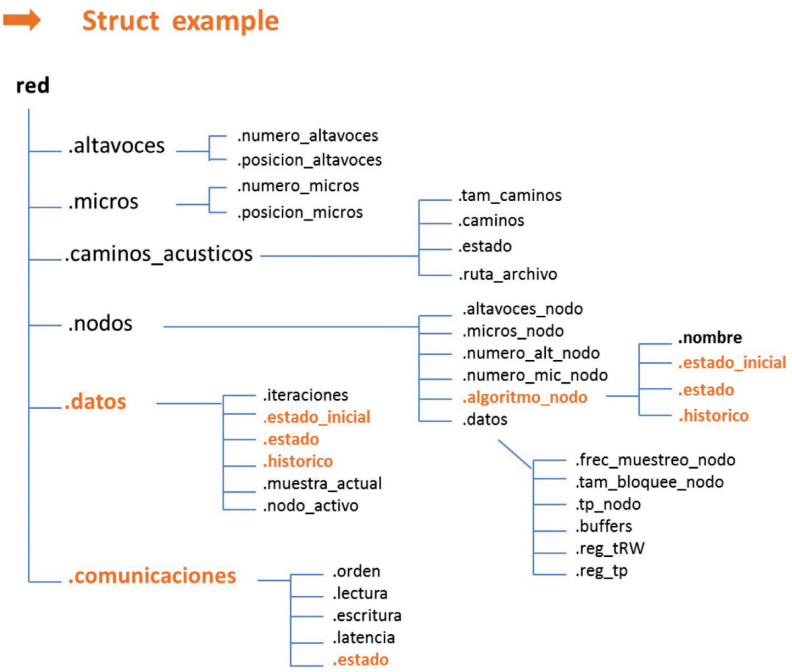

Figure 2. Example of the type of structure used in the simulator.

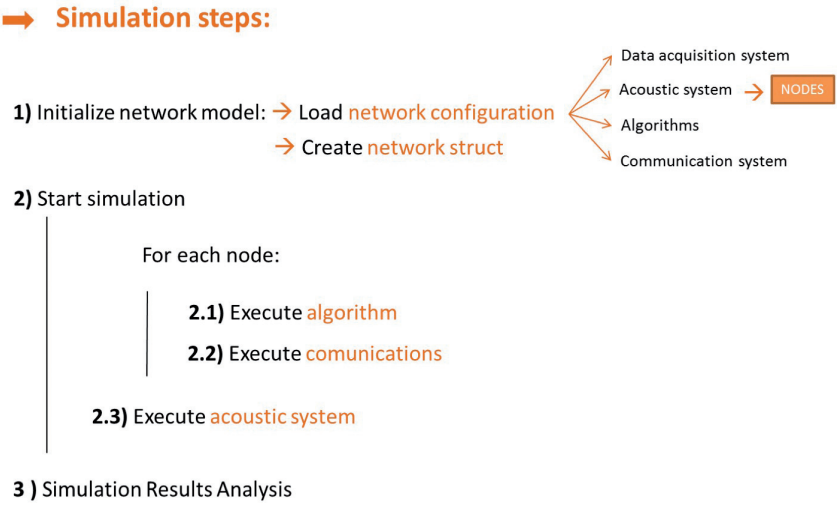

Figure 3. Steps to perform the simulation of an acoustic network model.

- Initial state: Field containing the initial values of the variables in the previous field.
- Historical: Field to store the variables that want to analyse.

A complete description of all the fields is out of the scope of this paper, but an example of one of the data structure used in the tool is shown in Figure 2.

Finally, Figure 3 shows the steps to run a complete simulation. The tool runs the algorithms at each node and simulates if there are latencies in the information exchange among the nodes. Once the signals have been processed by the algorithms and have been sent to the loudspeakers, the acoustic system generates the signals captured at the microphones locations. In this way, the tool can quickly perform different simulations for different applications by modifying the appropriate parameters in the initial configuration.

2.3. Simulation of the algorithms

The algorithms can work sample-by-sample or grouped in blocks of samples. A particular algorithm reads values in certain structure fields, executes the operations and writes in certain structure fields. The nodes can execute any algorithm once it is adapted to the format defined by the structure.

The general definition of an algorithm consists of four parts. An example is shown in Figure 4:

1) Declaration statement: It states the function name, the needed input variables from the structure and the output variables that are going to be returned to the structure.

2) Reading: It reads the field "State" of both the algorithm node and the communications system (if any information exchange between the nodes exists), obtaining the values of the variables used by the algorithm.

3) Processing: In this section, the necessary operations for the proper performance of the algorithm are executed. Depending of the type, the algorithm can process one or several input variables (corresponding to the input signals or the signals captured by microphones or parameters sent by other nodes, etc.) and one or several output variables (corresponding to the signals reproduced by the speakers, parameters to send to other nodes, etc.).

4) Writing: After processing and similarly to the reading process, the new values of the variables must have been stored in the state field of both the algorithm of each node and the communications network.

2.4. Simulation of the communication system

The communication system performs two types of operations in the simulator. On the one hand, it is possible to simulate latencies (or other effects) in the transmission of information to the nodes by delaying the input signals a certain number of samples at each node. On the other hand, if collaborative algorithms are used, the system allows to exchange information among the different nodes. In this case, the network topology explains how the nodes must collaborate stating for each node which nodes can read data from it and in which nodes it can write data. This information, along with the nodes that collaborate and their order in the network, must be introduced in the configuration of the communications network. Depending on the topology, the network will be responsible for sending that information to the corresponding node as well as introducing errors or delays, if needed.

2.5. Simulation of the acoustic system

Acoustic systems are formed by the number of actuators (loudspeakers) and number of sensors (microphones) along with the impulse response of the acoustic paths between them, that is, the acoustic system defines an enclosure and the position where the loudspeakers and microphones are placed within it. The acoustic system must work sample-by-sample in the time domain in contrast to the algorithms that can be run sample-by-sample or in blocks of samples. However, the programming of the acoustic system is quite similar to the definition of the algorithms. It contains first the declaration statement of the function followed by a part where the necessary variables are read from the fields of the structure. Second, the processing where the signals reproduced by the loudspeakers are filtered by the acoustic paths. Finally, the signals captured by the microphones are stored in the structure. An example is shown in Fig. 5.

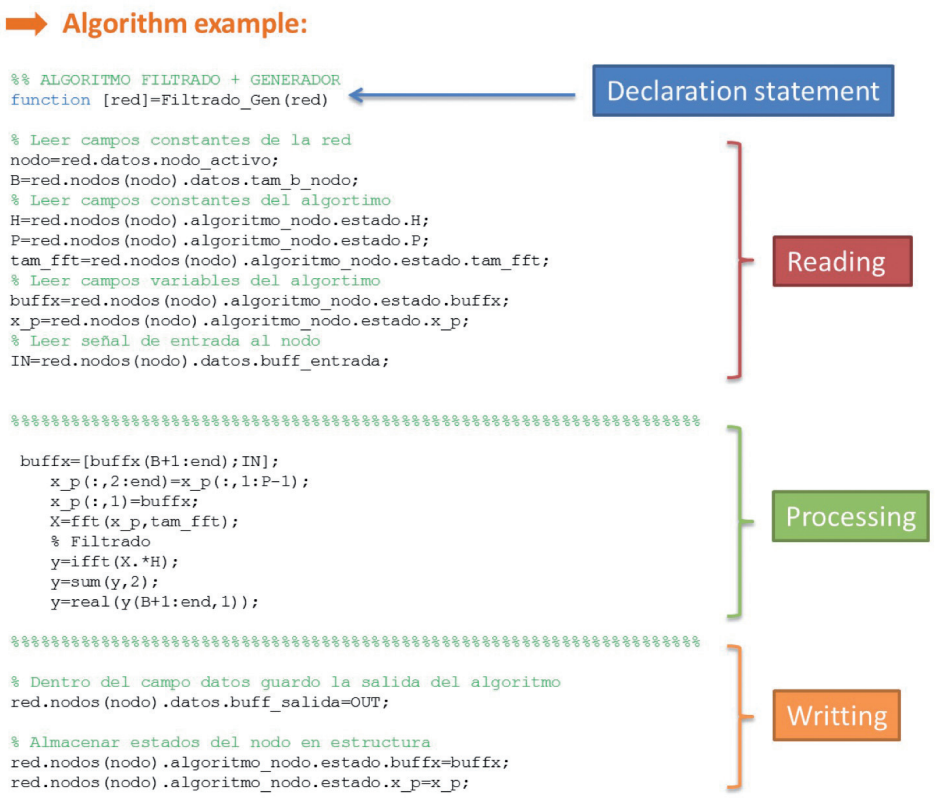

Figure 4. Example of an algorithm function.

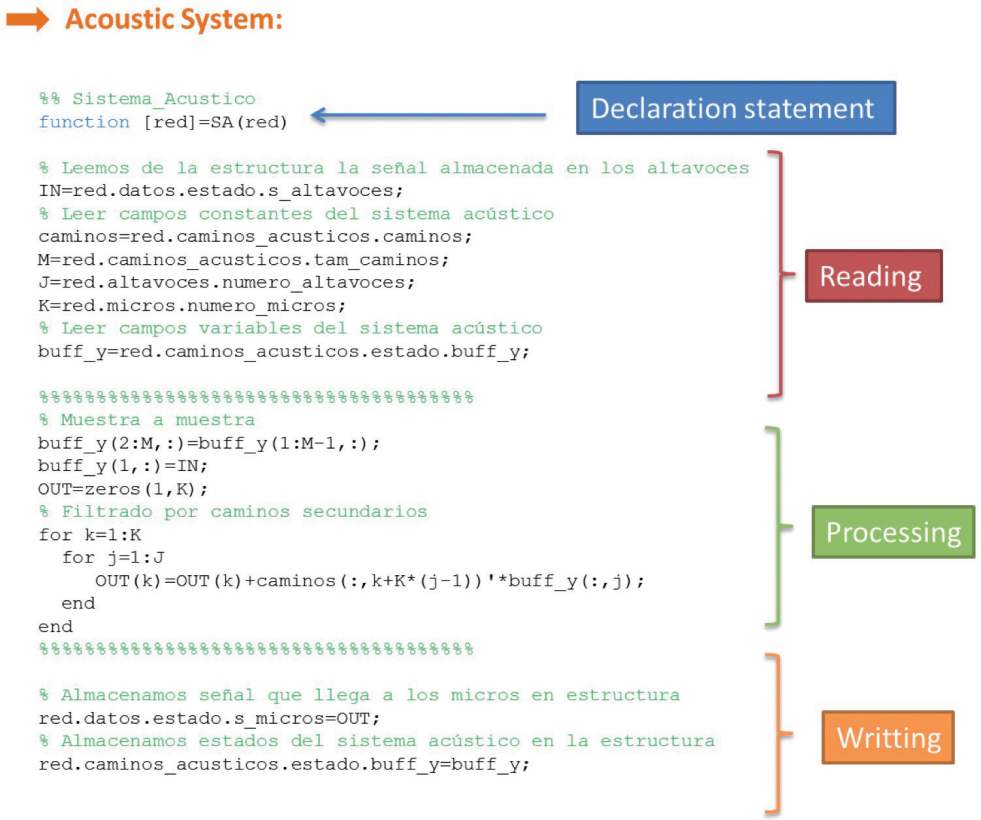

Figure 5. Function that models the acoustic system.

2.6. Analysis of the performance

The tool can analyse any existing signal or parameter in the system. Using a specific function, the tool stores in the structure only those signals captured by the microphones that have been previously selected as well as only those signals produced by the loudspeakers that have been previously chosen. This reduces the computational load and the stored memory in the cases of systems with a large number of microphones or speakers but that only need the signals captured or reproduced by some of them.

2.7. Limitations

At this moment the main limitation of the simulation tool is the lack of interaction with the user, but a graphical user interface is planned for the next phase of the project.

From the point of view of efficient programming in Matlab, the tool tries to use multiple dimension arrays (vectors, matrices) but the use of loops is unavoidable. Therefore, when a large number of nodes are used, both runtime and memory storage become prohibitive. In addition, working with different units of time increases the runtime.

Another issue is the resolution. The tool works with Matlab resolution but the idea is to simulate the main effects of AD/DA converters in the audio input/output respectively, as for instance the saturation and finite quantization of the samples. Finally, the tool does not support dynamic or variable acoustic systems.

3. WASNS over mobiles devices and wireless loudspeakers: current technology

In this section we explain how to develop a real WASN using mobile devices with ***Android*** operating system. We also indicate basic ***Android*** instructions to manage the Wi-Fi connection between mobile devices (smartphones and tablets) and the Bluetooth connection between devices and wireless speakers.

3.1. Programming the Android Operating System (O.S.)

Android has classes[1] for record and reproduce audio signals. The class *AudioRecord* can record audio signals from the device microphone and the class *AudioTrack* is used for audio reproduction. These classes work with signals without any kind of codification, that is, they use pulse coded modulated (PCM) signals. To explain how the above classes can record and reproduce simultaneously, it is necessary to introduce here the concept of *Thread* in ***Java***: a *Thread* is the smallest processing unit that allows the execution of concurrent tasks. Therefore, to record and reproduce different audio files at the same time, two *Threads* are needed.

Each *Thread* has two parts:

1. The initialization of the variables
2. The execution of the operations.

[1] Class: It defines an object specifyinng its properties and the operations that can perform.

An important point is that the initialization of the recording *Thread* has a higher computational complexity than the initialization of the reproduction *Thread*. This means that the operations of both do not start at the same time. To solve this problem the functions ***wait()*** and ***notify()*** are used. When the reproduction *Thread* finishes its initialization, it uses ***wait()*** and remains blocked until the recording *Thread* finishes its initialization and uses ***notify()*** to awake it.

3.2. Bluetooth communication

The Bluetooth protocol is used to connect a mobile device (smartphones or tablets) to a wireless loudspeaker. In order to understand the procedures of *Bluetooth* connections, it is necessary first to talk about the *Bluetooth* profiles. A *Bluetooth* profile is a service that the devices are able to use for the communication. There are a lot of profiles, but in this project only two of them are relevant:

1. A2DP profile: Advanced Audio Distribution Profile. It defines how multimedia audio is streamed between two devices.

2. HSP profile: Headset Profile. It provides support for the popular *Bluetooth* headsets to be used with mobile devices.

Android has two classes, *BluetoothA2dp* and *BluetoothHeadset*, to manage these two profiles. Thanks to these classes the process of connection can be done successfully. This process has two steps:

1. Pair[2] the chosen device.

2. Connect the chosen device.

If the device is already paired, the step 1 is not needed, but to connect two devices, they always must be previously paired.

The connection between a mobile device and a wireless speaker under A2DP makes the sound to be reproduced by the wireless speaker instead of by the local speakers embedded in the device.

3.3. Wi-Fi communication

The Wi-Fi protocol is used to connect the mobile devices (smartphones or tablets) among them. To perform a Wi-Fi link between two devices there are three steps:

1. Searching and connection of a network.

2. Searching of devices inside the chosen network.

3. Connection among devices with a TCP socket[3]. This last step is needed to assure a safe and reliable communication.

At the third step the ServerSocket and Socket classes are used. One device will act as a server and the other as a client, but only during the connection process. Assume that devices A and B act as server and client respectively. Device A will use the ***accept()*** method of the ServerSocket class to enable listening to connection requests from the client. This method is blocking, meaning that until a connection is not established, the following line of code is not executed. Moreover, device B will perform the connection request by the Socket class, which also blocks the code. Once the connection is established, both devices will have the same attributes for sending and receiving information.

Regarding the synchronization among the connected devices, it needs to be as tight as possible in order to start the reproduction on one device and the recording on the other one simultaneously.

Regarding the synchronization among the connected devices, it needs to be as tight as possible in order to start the reproduction on one device and the recording on the other one simultaneously. The procedure is shown in the diagram in Figure 6. Once the connection is established, both devices get their own time reference and stored them in **tm** and **ts** respectively. The master then executes the initialization of its variables and notifies the slave, waiting till the slave completes its initialization process and alerts to the master. At that time the master gets a new temporary reference **tm1** and transmits to the slave the difference **(tm1-tm)**. The slave then takes his own time reference **ts1** and calculates the delay with respect to the master as **(ts1-ts) - (tm1-tm)**. From this moment, both devices wait a time fixed by method *Thread.sleep(millisec)* where millisec is the number of milliseconds that each device has to wait to get synchronized, as is shown at the bottom of Figure 6.

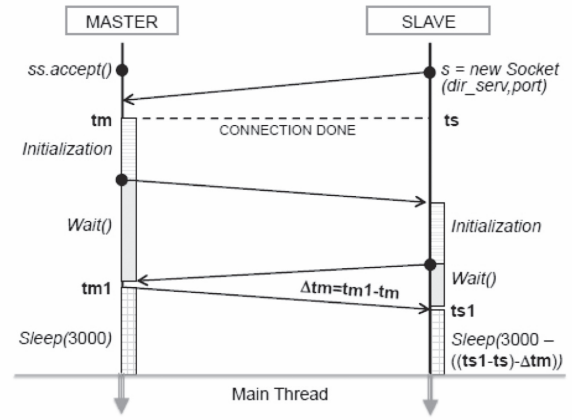

Figure 6. Procedure to synchronize two devices in Wi-Fi connections.

3.4. Technology limitations

At this point, we can divide the current limitations in three groups:

1. *Bluetooth* limitations:

   - The maximum distance between devices is 10 m.

   - One device can establish only one connection at the same time and the process of connecting and disconnecting is quite time consuming.

[2] Pair: It is a process to do safer the communication *Bluetooth*, but to pair does not mean to connect. In fact, a single device can be paired to several devices.

[3] Socket: It is a data communications endpoint for exchanging data between processes executed within the same host operating system.

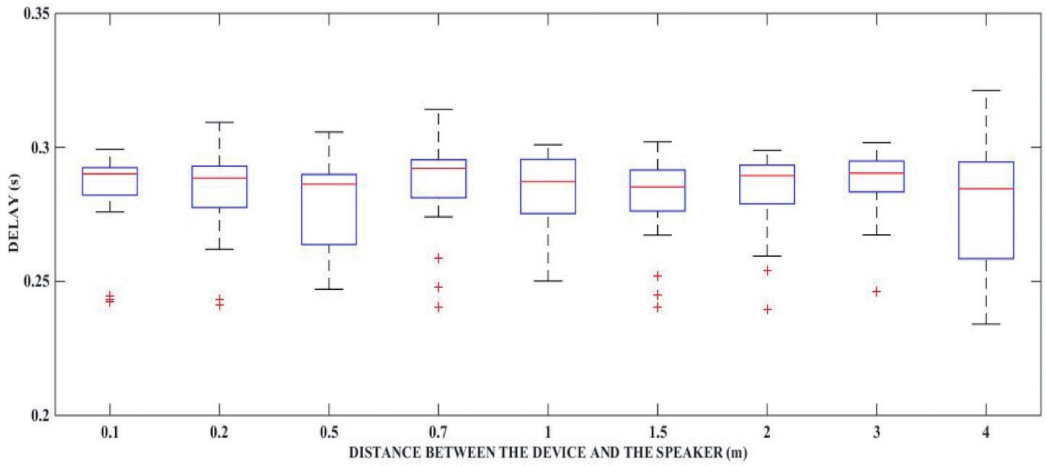

Figure 7. Distribution of O.S. delays in Bluetooth connections with respect to the distance between devices.

- The time spent by the O.S. to reproduce an audio file through a wireless loudspeaker is different at each reproduction. Figure 7 shows the distribution of the reproduction delay for different distances between the device and the loudspeaker. For each distance the delay has been obtained 20 times. It can be appreciated that the Bluetooth connection presents a mean delay (red line) below 0.3 ms independently of the distance. Unfortunately, the expected delays are spread between 0.23 and 0.33 s and cannot been predicted.

2. *Wi-Fi* limitations:

- The connection between devices must be done through a common router, which has to provide high transmission rates and support all the data traffic.
- As in *Bluetooth*, the delay is not constant, although it can be controlled as explained above. However, the expected delay is much smaller than in Bluetooth, around 5 ms.

3. ***Android*** limitations:

- The vector and matrix management is very inefficient.
- The processing times are not constant. Moreover, similar devices of the same model, with the same hardware and software (O.S.) can present very different processing times.

4. Conclusions

In this work, a simulation tool that allows the development of sound field control applications over distributed networks has been presented. The main advantage of this tool is the simplicity and quickness to simulate different systems. The tool allows us to create nodes with an arbitrary number of microphones and speakers and to execute any algorithm both in the time and the frequency domain, and both working sample-by-sample and grouped in block of samples. The simulator executes the acoustic system sample-by-sample independently and allows for modelling of both the behaviour of the communications system simulating the possible problems and the data acquisition performed in real time applications. Finally, the proposed tool allows to analyze any signal or parameter in the nodes, acoustic system or communication network, hence, at any point in the system. Moreover, the development of a real WASN programmed in Android and the basic instructions needed to manage Wi-Fi and Bluetooth connections between devices have been explained.

Acknowledgments

This work has been supported by European Union ERDF and Spanish Government through TEC2012-38142-C04 project, and Generalitat Valenciana through PROMETEOII/2014/003 project.

References

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, "A survey on sensor networks", Communications Magazine, IEEE, 40 (8) 102–114 (2002).
[2] K. Martinez, J.Hart, R.Ong, "Environmental sensor networks", Computer, 37 (8) 50–56 (2004).
[3] D. Puccinelli and M. Haenggi, "Wireless sensor networks: applications and challenges of ubiquitous sensing", Circuits and Systems Magazine, IEEE, vol. 5, no. 3, pp. 19–31, 2005.
[4] M. Maroti, G. Simon, A. Ledeczi, J. Sztipanovits, "Shooter localization in urban terrain", Computer, 37 (8) 60–61 (2004).
[5] W.-P. Chen, J. Hou, L. Sha, "Dynamic clustering for acoustic target tracking in wireless sensor networks", IEEE Trans. Mobile Comput., 3 (3) 258–271 (2004).
[6] A. Bertrand, "Applications and trends in wireless acoustic sensor networks: A signal processing perspective", in Communications and Vehicular Technology in the Benelux (SCVT), 2011 18th IEEE Symposium on, pp.1–6 (2011).

[7] A. Bertrand, S. Doclo, S. Gannot, N. Ono, and T. van Waterschoot, "Special issue on wireless acoustic sensor networks and ad hoc microphone arrays," Signal Processing, 107, 1-3 (2015).
[7] C. Lopes, A. Sayed, "Incremental adaptive strategies over distributed networks", IEEE Trans. Signal Processing, 55 (8) 4064–4077 (2007).
[8] N. Bogdanovic, J. Plata-Chaves, K. Berberidis, "Distributed incremental-based LMS for node-specific parameter estimation over adaptive networks", in Proceedings of 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 5425–5429 (2013).
[9] M. Ferrer, M. de Diego, G. Piñero, A. Gonzalez, "Active noise control over adaptive distributed networks", Signal Processing, 107, 82-95 (2015).
[10] H. Kim, S. Lee, J.-W. Choi, H. Bae, J. Lee, J. Song, and I. Shin, "Mobile maestro: Enabling immersive multi-speaker audio applications on commodity mobile devices", in Proc. 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14), 277–288 (2014).
[11] C.A. Kardous, P.B. Shaw, "Evaluation of smartphone sound measurement applications," The Journal of the Acoustical Society of America, 135 (4) 186–192 (2014).
[12] Matlab online documentation, http://es.mathworks.com/help/matlab/structures.html

Biographies

Christian Antoñanzas received the degree in Sound Technician from the ESCIVI in Andoain (Guipúzcoa) in 2005. He received the degree in Technical Telecommunications Engineering, specialising in Sound and Image from the Public University of Navarre in 2009. In the next year, he cursed a Master Degree in Acoustic Engineer in the Polytechnic University of Valencia (UPV). Besides, in 2011 he takes a Specialization Course in Business Consulting in the Public University of Navarre doing the practicum in IECISA. For 2 years he has been working as an Acoustic Engineer in privates companies of the audio-visual sector until in June he joined to the Audio and Communications Signal Processing Group at iTEAM as research technical for DisCoSound project thanks to a PhD grant from the Ministry of Research, Development and Innovation. His research is focused on the study of distributed adaptive algorithms to develop multi-channel sound control systems.

Juan Estreder Campos was born in Valencia, Spain, in 1991. He started to work at the Audio and Communications Signal Processing Group (GTAC) for his Master Thesis titled "Sound-field Controller over Acoustic Nodes through Distributed Signal Processing", which he defended at the Telecommunication Engineering School of the Polytechnic University of Valencia (UPV) on October 2014. Since December 2014 he works as a developer of sound control applications over a network of Android-based devices under the research project DISCOSOUND.

Gema Piñero received the M.Sc. degree in telecommunications engineering from the Polytechnic University of Madrid, Spain, in 1990, and the Ph.D. degree in telecommunications engineering from the Polytechnic University of Valencia, Spain, in 1997. She is currently an Associate Professor in the Department of Communications of the Polytechnic University of Valencia and a researcher within the Audio and Communications Signal Processing group (GTAC) of the Institute of Telecommunications and Multimedia Applications (iTEAM) of the same university. Since 1990 she has participated in several research projects in the areas of array signal processing, multi-channel sound systems, wireless communications and sound quality evaluation Particularly she has leaded research projects on sound quality evaluation for the automotive industry, and participated in projects on 3G and 4G wireless communications supported by public and private funding (Spanish Government, Regional Government, Telefonica, NVIDIA). She has published more than 80 contributions in international journals and conferences. Her current research interests include coordinated and space-time multi-user techniques in wireless communications systems, and distributed signal processing algorithms for wireless acoustic sensor networks. She is a senior member of the IEEE and founding member of the Spanish Association for Research and Teaching UNIVERSITAS.

Miguel Ferrer Contreras graduated in Telecommunications Engineering in the year 2000 at the Universidad Politécnica de Valencia (UPV), Spain. He was collaborating with the Audio and Communications Signal Processing Group (GTAC) since a year before, performing a six months research stay in the Instituto de Investigación Aplicada al Automóvil, Tarragona, Spain (Automobile Applied Research Institute). Subsequently, he was awarded several grants offered both by the Communications Department of the UPV and by the Research, Development and Innovation Vice-Chancellery of the same university, enabling him to start his Doctorate studies and to collaborate in different research projects within the GTAC. During this period he has authored or co-authored over twenty papers related with signal processing in renowned journals and conferences. Since 2005 he works as an assistant lecturer in the Communications Department of the Polytechnic University of Valencia (UPV). His research activity is focused on the study of adaptive algorithms and its application to audio digital processing and noise active control, a subject about which he is developing his doctoral thesis.

María de Diego was born in Valencia, Spain, in 1970. She received the Telecommunication Engineer degree from the Universitat Politècnica de València (UPV) in 1994, and the Ph.D. degree in 2003. Her dissertation was on active noise conformation of enclosed acoustic fields. She is currently working as Associate Professor in digital signal processing and communications. Dr. de Diego has been involved in different research projects including active noise control, fast adaptive filtering algorithms, sound quality evaluation, and sound reproduction, in the Institute of Telecommunications and Multimedia Applications (iTEAM) of Valencia. She has published more than 40 papers in journals and conferences about signal processing and applied acoustics. Her current research interest includes multichannel audio signal processing, distributed processing, and sound quality improvement.

Alberto González Salvador works as Professor at the Universitat Politècnica de València, Spain. He attended the Universidad Politécnica de Catalunya, Spain, where he graduated in 1991 in Telecommunications Engineering with highest honours. In 1997 he was awarded a Doctorate (PhD), magna cum laude, from the Universidad Politécnica de Valencia. During 1995 he worked as a visiting researcher in the Institute of Sound and Vibration Research (ISVR) at the University of Southampton, United Kingdom. Currently, he is the head of the research group in Audio and Multimedia Digital Signal Processing. Alberto González has published over 100 papers in international technical journals and renowned conferences in the fields of signal processing and applied acoustics. As well as this, he has leaded twelve research projects and has collaborated in twenty more. He is a member of the senate of the Universitat Politècnica de València and serves as Dean of the Telecommunications Engineering School since June 2012, having previously been head of the Communications Department and assistant director of research since 1997. His current research interests include multichannel signal processing for communications and three-dimensional sound reproduction.