DIGITAL COMMUNICATIONS LEARNING TOOLS
gtTAL: Graphical Tool for Testbed-assisted Learning

José A. García-Naya, Héctor J. Pérez-Iglesias, Adriana Dapena and Miguel González-López
Department of Electronics and Systems, University of A Coruña, Spain
{*jagarcia, hperez, adriana, mgonzalezlopez*}*@udc.es*

Keywords: Digital communications, Multiple-antenna testbed, Testbed-assisted learning.

Abstract: We introduce gtTAL, a graphical tool implemented on top of a distributed multilayer architecture which is specifically suitable for multiple-antenna hardware testbeds. gtTAL helps in teaching digital communications by allowing interaction with the hardware testbed at an abstraction level suitable for undergraduate students. Instead of using the low-level interfaces provided by hardware manufacturers, the multilayer software architecture supplies a high level interface access for testbeds, releasing students from the necessity of knowing low-level details of the hardware to start to practice with it. Therefore, they can easily test algorithms without developing a new program from scratch, speeding up the time needed for both the implementation and the debugging tasks. Indeed, the multilayer software architecture allows learning how to deal with real-world digital communication systems at different abstraction levels, varying from the lowest level software running in real-time in DSPs or FPGAs, to the highest level software like gtTAL. These three elements: hardware testbed, multilayer software architecture and graphical tool (gtTAL), constitutes what we termed *testbed-assisted learning*.

1 MOTIVATION

In wireless digital communication courses is common to use computer simulations to illustrate the theoretical concepts presented to the students. Unless very complex simulations are carried out, the most typical simulations in graduate courses show the corresponding results under controlled and ideal conditions. Thus, very important effects introduced by hardware elements are often ignored, like for example those caused by the antennas, by the D/A and A/D converters or by the radio-frequency (RF) amplifiers. Some of these effects can only be well understood if the students experience the problem by themselves. For this reason, computer simulations are only useful as an starting point in understanding the key concepts of modern wireless digital communications. However, computer simulations do not allow to study and understand very important implementation issues present in real-world transceivers.

There is a variety of multi-antenna testbeds that provides many educational opportunities (Rao et al., 2004), allowing the students to learn, step by step, all signal processing stages involved in the generation of the signals to be transmitted through the antennas. This transmit signal processing chain is very similar to that usually found in computer simulations. However, when the students have to implement the corresponding signal processing blocks of a real-world receiver, they find several important differences. Some of them are caused by implementation impairments, i.e. frequency and phase noise or non-linear distortions caused by RF power amplifiers. Other differences are inherent to the fact that transmitter and receiver are situated at different physical locations.

Consequently, both time and frequency synchronization steps have to be carried out prior to any other operation. Also, some operations like filtering, decimation and channel estimation (instead of perfect channel knowledge) are mandatory. In contrast to computer simulations, the sources of error are now *out of control*, which on one hand makes analyzing the results more difficult but, on the other hand, the students have to deal with a more realistic problem. It is important to stress that some effects can also be seen, and later understood, if the whole system implementation is considered. For example, most of the theoretical multiple-antenna models consider the same average gain for all individual single-antenna links. However, testbed measurements show that this is not true and the average gain depends on the specific propagation seen by each of the transmit-receive

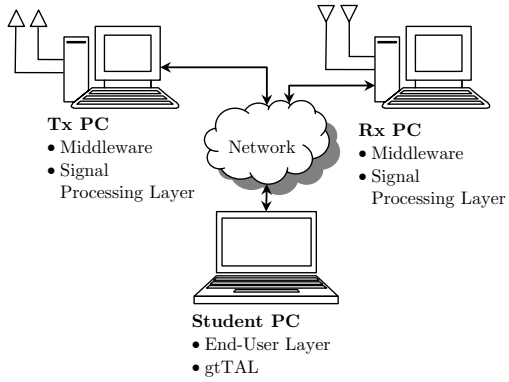

Figure 1: General organization of the testbed and the student PC running gtTAL.

antenna pairs.

In addition, being able to implement and test algorithms in a testbed requires more multi-discipline skills than the theoretical approach. For instance, implementing a time synchronization algorithm in a testbed requires to deal with low-level details of the testbed hardware components, as for example the type and format of the A/D converter outputs. Also, it requires skills in instrumentation equipment usage (e.g. oscilloscopes, spectrum analyzers, network analyzers, etc.). In summary, *testbed-assisted learning* provides a much wider perspective of communications engineering to the student.

The rest of this paper is organized as follows. 2 shows the main contributions provided by the so-called *testbed-assisted learning*. 3 presents the distributed multilayer software architecture of the testbed, 4 describes the testbed hardware components, and 5 details the architecture layers. 6 illustrates how to use gtTAL in a digital communications lesson. Finally, 7 is devoted to the conclusions.

2 BENEFITS OF TESTBED-ASSISTED LEARNING

Some of the most relevant researchers in the field of digital wireless communications (e.g. (Rao et al., 2004)) claim that students involved in testbed implementations experience a work that is more time-consuming, but more rewarding than single-discipline tasks. For instance, rarely will a communication theory student need to spend time understanding the impact of I/Q imbalances, while a student working on a testbed has to take into account such effects. Also, students working on system implementations notice that the testbed approach is more detailed and comprehensive than only theory and computer simulations. Therefore, the testbed approach leads to a greater satisfaction in seeing a fully functional testbed transmitting and acquiring real-world signals.

Being involved in testbed development tasks forces the student to work in a multidisciplinary environment (e.g. computer engineers and electrical engineers use to work together in the development of such testbeds). The student learns how to work in a team and also how to discuss with senior members about different issues found during the implementation.

During the last years, different general-purpose multiple-antenna testbeds have been constructed to evaluate the performance of diverse signal processing techniques and/or standards (e.g. (Caban et al., 2006; Borkowski et al., 2006)). At a first glance, one may think that undergraduate students can participate in the development of testbeds but, in fact, only postgraduate students with high expertise are involved in such teams. Unfortunately, setting up and later develop software allowing to transmit, acquire and properly process signals involves cumbersome low-level programming to access the hardware, making difficult to test new methods which allows the students to start to interact with testbeds (Rupp et al., 2007). Due to this reason, it is convenient to add a mechanism to the testbed that allows to access it at different levels of abstraction. This means that a student starting to implement his first algorithms should access the testbed at a higher level than another student who is prepared to deal with more low-level details of the testbed. Consequently, such a mechanism allows the students to focus exclusively on the development and proof techniques, releasing them from the task of low-level programming.

Our aim is, thus, to provide a mechanism that allows even undergraduate students to access the testbed at an abstraction level similar to that of computer simulations. Once the student acquires skills enough to deal with lower level details, this mechanism should also permit using the testbed but still hiding *even lower* level details. The understanding of a multi-layer scheme, where each layer deals with specific problems at a different abstraction level, clearly leads to enforce the knowledge of the students.

To the knowledge of the authors, a multitude of universities as well as public and private research centers have been investing a lot of efforts in setting up testbeds with research purposes. However, very few works (i.e. (Rao et al., 2004)) consider the possibility of taking advantage of the educational possibilities offered by testbeds. Therefore, we introduce the term "testbed-assisted learning" that consists in involving

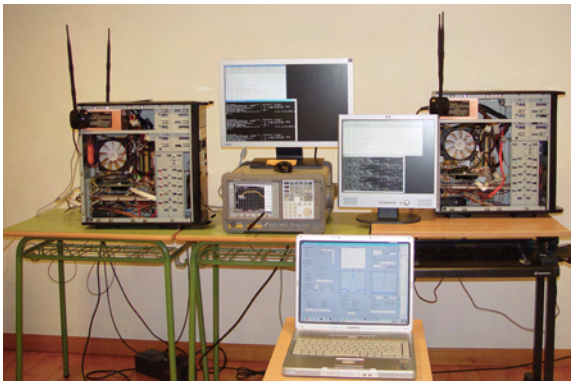

Figure 2: Testbed picture showing the Tx PC, the Rx PC, and gtTAL running on a laptop.

testbeds in the learning process. This approach is not restricted to the wireless communications field but our testbed and most of our research knowledge comes from this area. Consequently, we will restrict our discussion to this field.

3 TESTBED OVERVIEW

In Figure 1, the general organization of our testbed is shown. The testbed is hosted by two ordinary PCs (see Figure 2), one for the transmitter (referred to as Tx PC), and another one for the receiver (called Rx PC). Figure Figure 3 shows the block diagram of the entire system. Three main parts can be distinguished (from bottom to top): the testbed hardware that allows us to transmit discrete-time signals over multiple antennas at 2.4 GHz; the multilayer software architecture that makes the hardware accessible to end users (in the following, we consider the students also as end users of the testbed); and finally, the graphical tool gtTAL, which is implemented on top of the multilayer software architecture.

The lowest level layer (i.e. the middleware) is required to be installed in the same computers as the testbed hardware is hosted, but the other two layers can be installed in any other available PCs. However, the configuration we present here simplifies the set-up because all software needed is installed in the Tx PC and in the Rx PC. Only the user layer is installed together with gtTAL in the student PCs (see Figure 1).

The multilayer software architecture presented above provides a high abstraction level, which allows to implement user applications without knowing the testbed hardware details. For instance, in this paper we present gtTAL, a graphical tool designed to explain the so-called Alamouti code (Alamouti, 1998), constituting one of the simplest and most known space-time block codes frequently used to introduce multiple-antenna systems to the students. Such graphical application (detailed below) constitutes the fundamental tool to enable the teacher to show the main effects caused by real-world transmissions through the testbed. In this work we will restrict our approach to describe the benefits provided by the use of such tool in an academic environment, but the testbed plus the software architecture also enables many teaching possibilities for more advanced students (e.g. accessing the testbed at specific layers).

4 TESTBED HARDWARE DESCRIPTION

A picture of the Testbed PCs (Tx PC and Rx PC) is shown in Figure 2. The hardware of the testbed is entirely based on Sundance Multiprocessor Ltd (see the bottom of Figure 3). The transmitter is based on a PCI carrier board SMT310Q and the SMT365, a basic processing module equipped with an FPGA, a DSP, memory buffers and two buses capable to sustain a transfer rate of 400 MB/s. The basic processing module is directly connected to the data acquisition module (DAQ module), the SMT370. It contains a dual D/A converter with dedicated memory accessible at the same speed of the D/A converter. The DAQ module also has two A/D converters. Finally, the DAC module is connected to the RF front-end module, the SMT349, which performs up and down conversion operations from 70 MHz to 2.45 GHz with 16 MHz of maximum bandwidth. The receiver employs the same configuration as the transmitter (see the bottom of Figure 3) but incorporating a buffer memory module, the SMT351, allowing to store in real-time the data acquired by the A/D converters to be later transferred to the Rx PC.

In order to show the advantages derived from the use of a multilayer software architecture, we will explain a frame transmission step by step (see Figure 4).

- Once the symbols to be transmitted have been generated at the graphical tool, a function is called passing to it the corresponding symbol vectors (one vector per transmit antenna).
- These symbols are sent to the signal processing layer (TXPROC) through the user layer, where they are converted to pass band signals that are subsequently sent to the middleware.
- When both the Tx PC and the Rx PC are ready to complete a transmission, the signals are passed to the testbed hardware to be transmitted through the antennas.

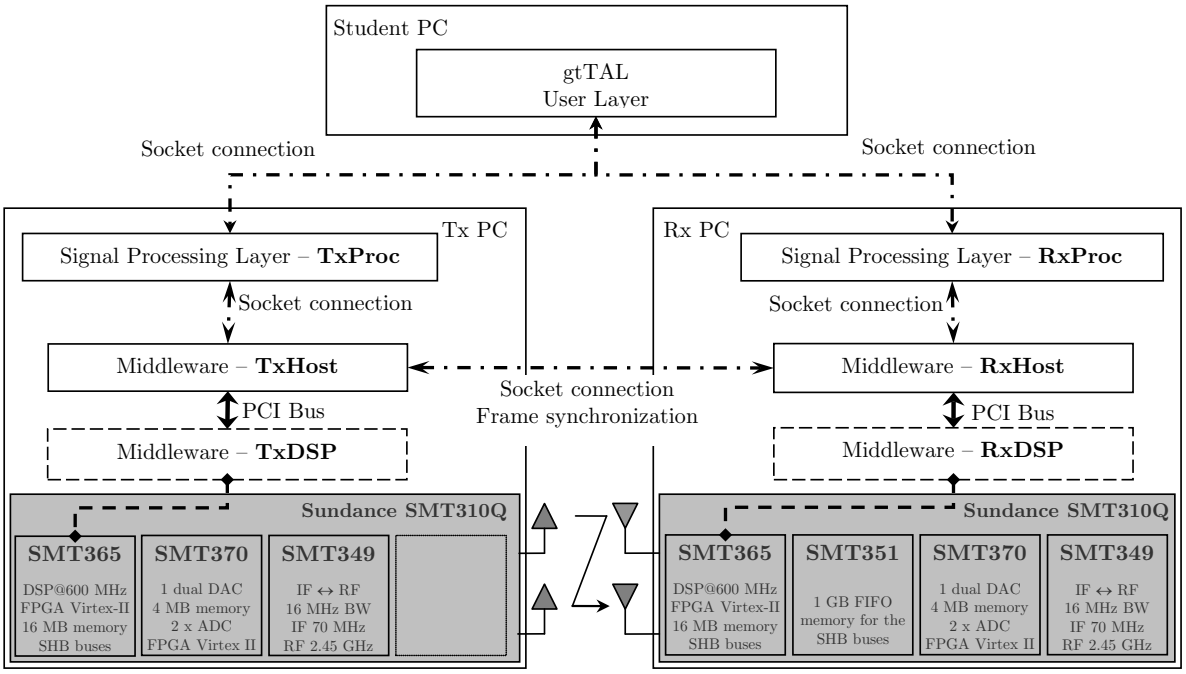

Figure 3: Platform Scheme.

- At the receiver side, the middleware stores the signals into the hardware buffers and then they are forwarded to the receiver signal processing layer (RXPROC).
- Finally, the acquired signals are forwarded to the user application through the user layer, completing the entire process.

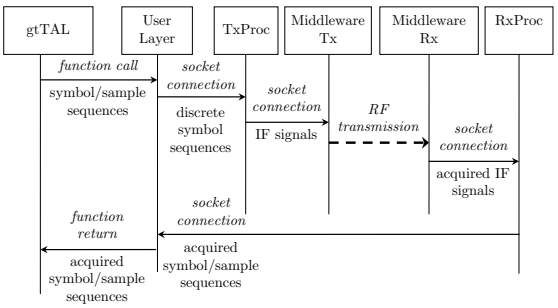

Figure 4: Example of a transmission using gtTAL, the distributed multilayer software architecture, and the testbed hardware.

The user layer allows configuring several parameters. For example, the signal processing layer can perform time and frequency synchronization or acquire non-synchronized signals, also raw data or already demodulated discrete symbols can be acquired.

5 ARCHITECTURE LAYERS

This section describes the three layers developed by us to provide high-level access to the testbed (see Figure 3): the user layer, the signal processing layer and the middleware layer.

5.1 User Layer

The user layer interacts with the user application (the graphical tool in this case) by using a simple function implemented in MATLAB® (any other software implementing socket connections is also valid). Its main task consists in sending to the signal processing layer the symbols to be transmitted plus the necessary parameters, at the transmitter side. In the same way, the user layer receives the acquired symbols, being noticed if any error occurs.

The main target of the user layer is making the rest of the layers accessible to the high level applications, taking into account the type of development environment they use. For this reason, the user layer is jointly executed with that application (see Figure 3).

5.2 Signal Processing Layer

The signal processing layer is network-connected with both the user and the middleware layers (see Fig-

ure 3). It provides remote access and makes them independent with respect to the other layers. This layer consists of two different processes that carry out the signal processing operations needed to link the user and middleware layers. The first process (TXPROC) receives the symbol vectors from the user layer and performs the up sampling, pulse-shape filtering, I/Q modulation and frame assembling operations in order to generate the IF signals that will be sent to the middleware. Similarly, the second process (RXPROC) waits for the acquired signals from the middleware and performs the time and frequency synchronization operations followed by the I/Q demodulation, filtering and down sampling. The resulting vectors are sent to the user layer.

5.3 MIDDLEWARE LAYER

The middleware concept constitutes a great leap forward in multiple-antenna testbed technology, making the testbed hardware accessible through ordinary network connections. This layer fills the gap between the testbed hardware and the signal processing layer, allowing discrete-time signals to be transferred through the PCI bus and making possible the synchronization between the Tx PC and the Rx PC using a network connection.

The middleware architecture is split into two different sub-layers (see Figure 3). The top sub-layer is responsible of establishing the network connections between the transmitter and the receiver, and with the higher layer (the signal processing layer). The bottom sub-layer corresponds to the testbed hardware configuration and control software.

The middleware is constituted by four different processes. The first two (`TxHost` and `RxHost`) implement the so-called top sub-layer and run, respectively, on the Tx PC and the Rx PC. They are implemented in standard C++ language and use sockets to establish the necessary network connections: one between `TxHost` and `RxHost` processes (used to synchronize the transmitter and the receiver, thus the receiver knows when the signal acquisition process has to start); another one, established between the TXHOST process and the Tx signal processing layer; and, finally, another one between the RXHOST process and the Rx signal processing layer. The remaining two processes are the transmitter and the receiver processes that run on their respective Digital Signal Processors (DSPs) available in the testbed hardware. They implement the so-called bottom sublayer. The transmitter DSP process (TXDSP) performs data transfers through the PCI bus jointly with the TXHOST process and configures and controls the hardware components at the Tx PC. In the same way, the RXHOST process and the DSP receiver process (`RxDSP`) are responsible of transferring the data through the PCI bus and, from the DSP side, controlling and configuring the testbed hardware components at the Rx PC.

6 A LESSON IN DIGITAL COMMUNICATIONS

Thanks to the abstraction level of our distributed architecture, it is very easy to devise a graphical tool for testbed-assisted learning (gtTAL). This tool helps in explaining basic concepts about wireless digital communication transceivers, including multiple-antenna systems. For instance, we have developed a first release of gtTAL oriented to explain Orthogonal Space-Time Codes (OSTBC), including the popular 2×1 Alamouti OSTBC (Alamouti, 1998).

The students should be familiar with some basic concepts in the field of wireless digital communications, such as modulation types (e.g. PAM, PSK, QAM), symbol rate and bandwidth, the concept of SNR (Signal-to-Noise Ratio), matched filtering, etc.

6.1 The 2×1 Alamouti OSTBC

Signal Model

Let us start with the explanation describing the base band model of the 2×1 Alamouti OSTBC. Two antennas are used at the transmitter side and only one antenna is employed at the receiver side (see Figure 5). As shown in Figure 5, the input binary data stream b_i is first mapped to the corresponding symbols, which are then split into two sub-streams s_1 and s_2. Each pair of modulated symbols $\{s_1, s_2\}$ is then transmitted during two consecutive time slots using the following strategy: during the first time slot, s_1 and s_2 are respectively transmitted through the first and the second antenna. During the second time slot, $-s_2^*$ is transmitted through the first antenna while s_1^* is transmitted through the second one[1].

Since the source symbols are sent through the antennas during two consecutive time slots, they experience different fading realizations h_1 and h_2 (see Figure 5), but the fading value is assumed to be the same during two time slots (i.e. a block fading channel with a duration of at least two channel uses). Hence, the

[1] The operator $(\cdot)^*$ denotes complex conjugation.

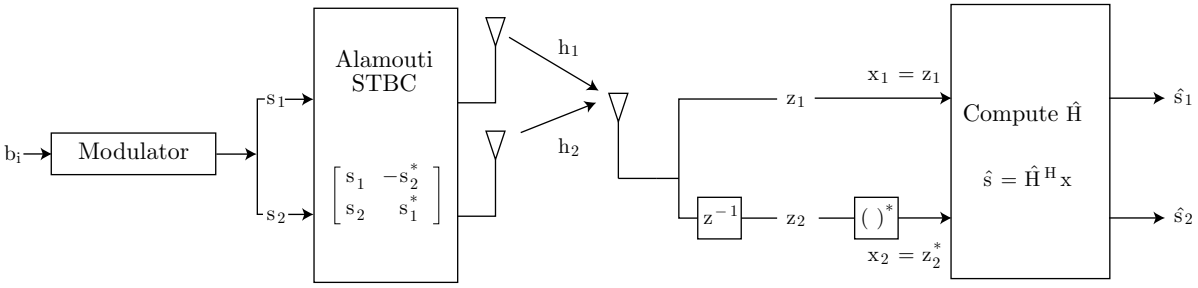

Figure 5: Alamouti OSTBC scheme.

signal received during the first time slot has the following form:

$$z_1 = s_1 h_1 + s_2 h_2 + n_1. \quad (1)$$

Given that the channel remains constant during two time slots, the observation in the second time slot is given by

$$z_2 = s_1^* h_2 - s_2^* h_1 + n_2. \quad (2)$$

In the expressions presented above, n_i denotes the additive white Gaussian noise (AWGN). By defining the vector of observations as:

$$\mathbf{x} = [x_1, x_2]^{\mathrm{T}} = [z_1, z_2^*]^{\mathrm{T}} \quad (3)$$

where the operator $(\cdot)^{\mathrm{T}}$ stands for the transposition. The relationship between the observation vector $\mathbf{x}$ and the source vector $\mathbf{s} = [s_1, s_2]^{\mathrm{T}}$ is given by

$$\mathbf{x} = \mathbf{H}\mathbf{s} + \mathbf{n} \quad (4)$$

where $\mathbf{n} = [n_1, n_2^*]^{\mathrm{T}}$ is the noise vector and $\mathbf{H}$ represents the (2×2) matrix obtained following the Alamouti coding scheme from the two channel coefficients h_1, and h_2 (mixing matrix):

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_1 \mid \mathbf{h}_2 \end{bmatrix} = \begin{bmatrix} h_1 & h_2 \\ h_2^* & -h_1^* \end{bmatrix} \quad (5)$$

It is interesting to note that $\mathbf{H}$ is unitary up to a scalar factor:

$$\mathbf{H}\mathbf{H}^{\mathrm{H}} = \mathbf{H}^{\mathrm{H}}\mathbf{H} = \|\mathbf{h}\|^2 \mathbf{I}_2 \quad (6)$$

where $\|\mathbf{h}\|^2 = |h_1|^2 + |h_2|^2$ is the squared Euclidean norm of the channel vector, $\mathbf{I}_2$ is the 2×2 identity matrix and $(\cdot)^{\mathrm{H}}$ denotes the Hermitian operator. Consequently, the transmitted symbols can be recovered up to scalar factor:

$$\hat{\mathbf{s}} = \hat{\mathbf{H}}^{\mathrm{H}} \mathbf{x}$$

where $\hat{\mathbf{H}}$ is a suitable estimate of the mixing matrix. As a result, this scheme supports maximum likelihood (ML) detection based only on linear processing at the receiver.

Channel Estimation Strategies

The performance of communication systems based on the Alamouti coding scheme strongly depends on the accurate estimation of the channel matrix $\mathbf{H}$. For this reason, this lesson is focused on supervised and unsupervised algorithms to estimate the mixing matrix. The standard way to estimate this matrix consists in utilizing pilot symbols (Budianu and Tong, 2001) known by both the transmitter and the receiver. Among the supervised methods, the Least Squares (LS) criterion (Haykin, 2001) constitutes a frequent starting point, given the simplicity of the resulting technique.

The main inconvenient caused by the use of pilot symbols is the energy spent during their transmission. Because pilot symbols do not convey data, the result is a loss in terms of spectral efficiency. This drawback can be avoided by using unsupervised approaches (also known as blind channel estimation methods). Blind Source Separation (BSS) algorithms can estimate the mixing matrix, $\mathbf{H}$, and therefore the realizations of the source vector $\mathbf{s}$, from the corresponding observations $\mathbf{x}$. The lack of *a priori* knowledge may limit the achievable performance, but makes blind approaches more robust to calibration errors (i.e. deviations from the assumed theoretical model) than conventional array processing techniques (Cardoso and Souloumiac, 1993). The best known BSS algorithms are the so-called joint approximate diagonalization of eigenmatrices (JADE), and FastICA (Bingham and Hyvärinen, 2000). More recently, specific algorithms for OSTBCs have been designed taking advantage of the specific structure of these codes (e.g. (Beres and Adve, 2007)).

6.2 Exercise

Our graphical software tool, gtTAL, allows testing either with complex-valued discrete symbol sequences

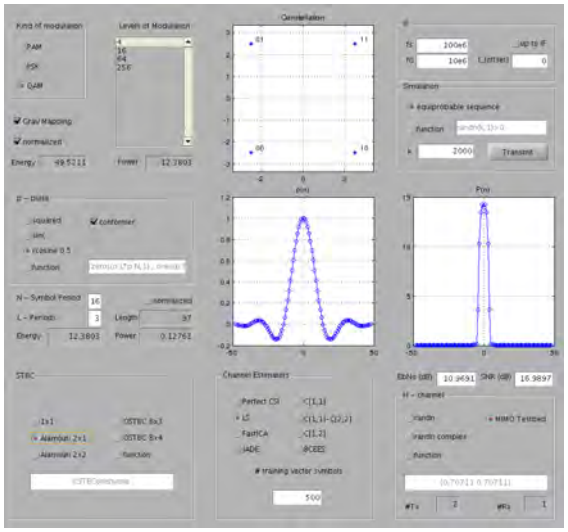

Figure 6: Screen shot of gtTAL main window.

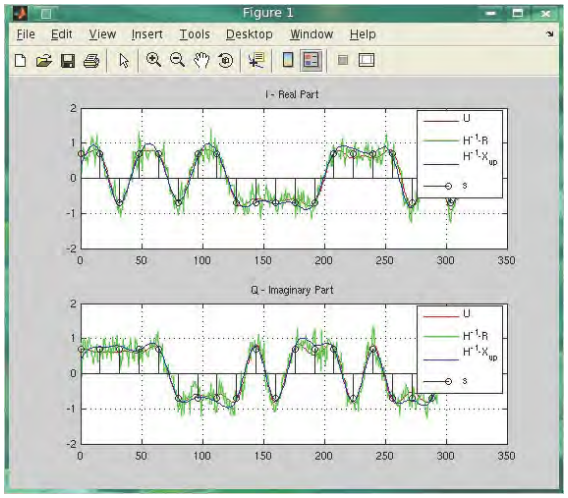

Figure 7: Result window showing the transmitted symbols (black) and signals (red), the acquired signals after the I/Q demodulation (green), and after the matched filter (blue).

or pulse-shaped signals, both base band or pass band. In all cases, the signals are transmitted using the testbed or through randomly generated channels (i.e. like in a conventional computer simulation). Several parameters can be modified such as the number of bits to be transmitted, or how they are generated (equiprobable source or using a function provided by the user); the modulation type (PAM, PSK or QAM), and the number of levels of the modulation; the number of samples for each symbol or the pulse-shape form to be used (square, root raise cosine or user-defined). Figure 6 shows the main window of gtTAL, used to introduce these parameters. Students can measure the performance of several channel es-

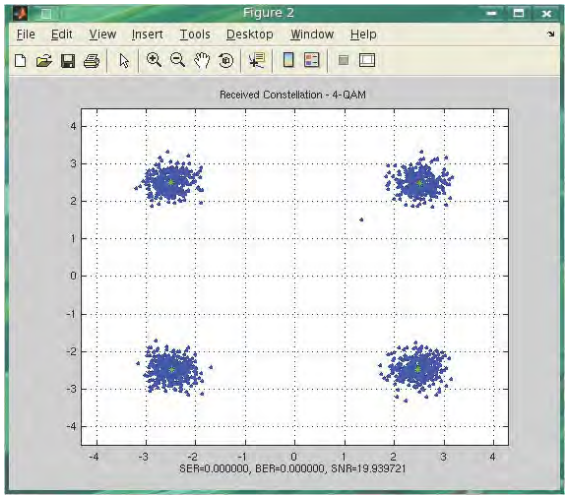

Figure 8: Symbol constellation corresponding to the acquired signals.

timation algorithms (Least Squares, Fast-ICA, JADE, etc.) and compare it with the performance obtained when perfect channel state information is available at the receiver side. In the latter case, it is required that the mixing matrix $\mathbf{H}$ be randomly generated using different distribution models (i.e. not using the testbed).

For example, the parameters introduced in the main window of gtTAL shown in Figure 6 allows to generate $k = 2000$ equiprobable bits mapped using the Gray code. The symbols are 4-QAM modulated, filtered using a root raised cosine pulse shape, and finally, the symbols are space-time coded using the Alamouti scheme. gtTAL displays the modulation constellation and the pulse shape.

After pressing the "Transmit" button, the sequence is transmitted over the testbed as it is shown in Figure 4, and the tool plots Figure 7 and Figure 8. Figure 7 shows the transmitted signal s_1 (another figure is plotted for s_2), the transmitted symbol values, the transmitted signal after the pulse-shape filter, and the acquired signals after the matched filter. Figure 8 shows the received symbol constellation. At the bottom of the figure, the estimated values for the symbol error ratio (SER), the bit error ratio (BER) and the signal-to-noise ratio (SNR) are displayed.

7 CONCLUSIONS

We have introduced gtTAL, a graphical tool to help in the teaching process of wireless digital communications. In its current release, gtTAL results very adequate to explain the basic concepts about space-time block codes for multiple-antenna transceivers. In par-

ticular, it includes the well known Alamouti space-time code. Also, it offers the possibility of using different channel estimation strategies, including supervised and unsupervised methods.

gtTAL can be used as a conventional graphical user interface to perform computer simulations, i.e. without needing any additional hardware component. However, its main potential resides on its ability to transmit and acquire signals by using a hardware testbed. To do so, gtTAL perfectly integrates with a distributed multilayer software architecture, that enables to access the hardware testbed avoiding low-level programming. In this sense, gtTAL represents a high-level tool to operate a hardware testbed. Thus, even undergraduate students can easily experiment with real data transmissions in realistic environments. This clearly contributes to increase their motivation as well as their personal reward in learning digital communications.

The hardware testbed, the multilayer software architecture, and gtTAL, constitute what we termed *testbed-assisted learning*.

ACKNOWLEDGEMENTS

This work has been funded by the Xunta de Galicia, the Ministerio de Ciencia e Innovación of Spain, and the FEDER funds of the European Union under the grants with numbers 09TIC008105PR, TEC2007-68020-C04-01, and CSD2008-00010.

REFERENCES

Alamouti, S. (1998). A simple transmit diversity technique for wireless communications. *IEEE Journal on Selected Areas in Communications*, 16(8):1451–1458.

Beres, E. and Adve, R. (2007). Blind channel estimation for orthogonal stbc in miso systems. *IEEE Transactions on Vehicular Technology*, 56(4):2042–2050.

Bingham, E. and Hyvärinen, A. (2000). A fast fixed-point algorithm for independent component analysis of complex valued signals. *International Journal of Neural Systems*, 10:1–8.

Borkowski, D., Brühl, L., Degen, C., Keusgen, W., Alirezaei, G., Geschewski, F., Oikonomopoulos, C., and Rembold, B. (2006). SABA: A testbed for real-time MIMO system. EURASIP Journal on Applied Signal Processing, 2006.

Budianu, C. and Tong, L. (2001). Channel estimation for space-time orthogonal block codes. In *IEEE International Conference on Communications, ICC 2001*, volume 4, pages 1127–1131.

Caban, S., Mehlführer, C., Langwieser, R., Scholtz, A. L., and Rupp, M. (2006). Vienna MIMO Testbed. *EURASIP Journal on Applied Signal Processing*, 2006, Article ID 54868.

Cardoso, J. and Souloumiac, A. (1993). Blind beamforming for non-gaussian signals. *IEE Proceedings F Radar and Signal Processing*, 140(6):362–370.

Haykin, S. (2001). *Adaptive Filter Theory*. Prentice Hall Information and System Sciences Series, 4th edition.

Rao, R., Zhu, W., Lang, S., Oberli, C., Browne, D., Bhatia, J., Frigon, J.-F., Wang, J., Gupta, P., Lee, H., Liu, D., Wong, S., Fitz, M., Daneshrad, B., and Takeshita, O. (2004). Multi-antenna testbeds for research and education in wireless communications. *IEEE Communications Magazine*, 42(12):72–81.

Rupp, M., Caban, S., and Mehlführer, C. (2007). Challenges in building MIMO testbeds. In *Proc. of the 13th European Signal Processing Conference (EUSIPCO 2007)*, Poznan, Poland.