An EEG (Bio)Technological System For Assisting The Disabled People

Dan-Marius Dobrea, Monica-Claudia Dobrea
Faculty of Electronics and Telecommunications, "Gh. Asachi" Technical University, Iasi, Romania

***Abstract*-One of the main goal of our research is the implementation of a real time brain-computer interfacing system able to interpret the EEG signal and, accordingly, to command different external devices. Our team proposes a new design for this kind of bio-instrumental systems, based on the concept of dynamic modularity of the entire system. The system is conceived as an independent, self-consistent unit that analyses and classifies in real-time different mental tasks for assisting persons with motor disabilities. In this paper we present some of the obtained results. At the end, this mobile unit (implemented with a PC104+ system) will be itself a modular, easily reconfigurable development platform dedicated to solve, in a more facile manner, different experimental classification paradigms.**

I. Introduction

Nowadays, there is a number of brain computer interface (BCI) systems that are presented in the literature, all of these representing the concepts and implementations of different research teams from the BCI field (for e.g., see [1], [2]).

Analyzing these BCI systems one could notice that they are designed only for a particular activity of the central nervous system. Moreover, each of them processes the particular activity in a particular way (using specific processing and classification methods), and then controls, in a certain manner, a device for one or several subjects. For these reasons, it was concluded that all these systems are not suited for systematic studies in the field. Thus, the implementation of a more general BCI system circumventing these drawbacks has become a necessity.

In addition, the impossibility of an impartial evaluation of the performances of different systems – due to the diversity of the employed techniques, to the structural design differences of the applied system, to the data sets, to the different investigated mental tasks etc. – represents another disadvantage.

The lack of a common vocabulary between different laboratories that are involved in this field was mentioned as well, [3]. In this respect, it was argued that the poor communication is generated by the lack of common terms and by the inexistence of a functional model in designing the BCI systems, [3].

For example, one of these systems – used by the Medical Informatics Department in the University of Technology Graz, Austria –, runs on a Pentium II IBM compatible platform and it has, as a central core used for software development, the Matlab with Simulink environment [1]. Basically, this system was intended to replace the old one "Graz-BCI" which had a series of drawbacks: the necessity of using simultaneously an external DSP processing kit (with all the existing inconveniencies: the need of writing C and assembly code for this specific device, the necessity of existence of a communication software module with the personal computer and so on), the necessity of format conversion for the acquired signal files for the *offline* analysis, the impossibility of using the same code for *online-offline* analysis, the static character of the system (the new system requirement was to run on a *laptop*) etc.

The authors of the last proposed system presented above consider the use of the Matlab™ with Simulink™ environment (a graphical environment tool used for application development) and of the Real-Time Workshop module (which allows the automatic generation of the developed application code) as being its major advantage [1]. Both, Simulink and the Real-Time Workshop module are parts of the Matlab environment.

Nevertheless, the same authors admit the necessity of writing the critical subroutines in C language (with an external compiler) followed, then, by their embedding in the Matlab environment [1].

One first attempt to standardization in the field of BCI systems was proposed in [3]. The authors suggested a general model for the BCI systems. Unfortunately, this model was a pure theoretical one and it did not succeed to impose itself.

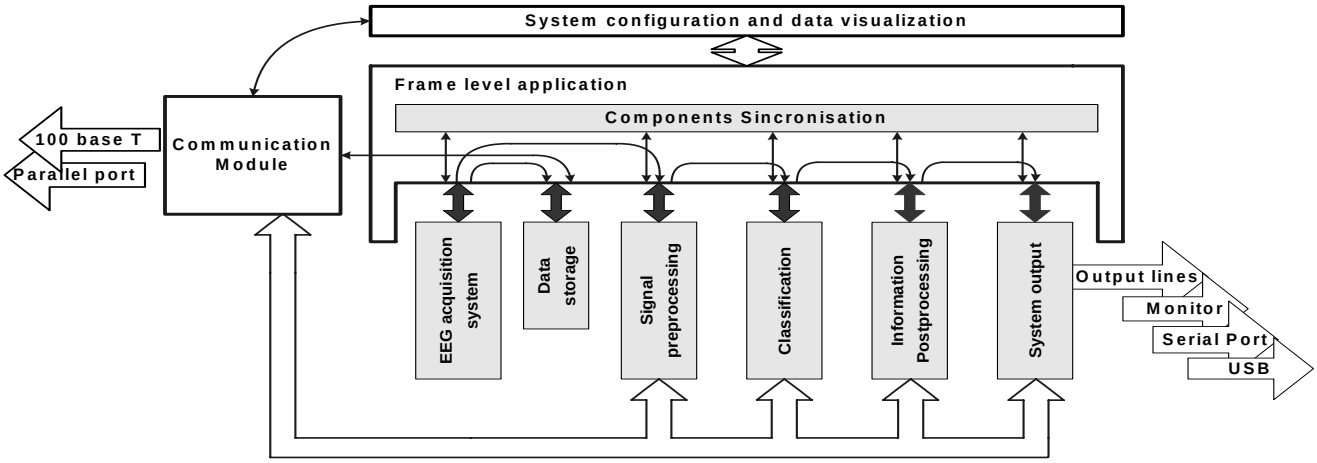

Figure 1. The schematics of the new system

In order to achieve a systematic evaluation and objective assessment between different applied methods, algorithms, operating protocols etc. another general BCI system, called BCI2000 [2], was proposed. This BCI2000 system is composed of four different modules that intercommunicate. Each module represents a functional block of the system. The inter-modules communication is realized by means of the TCP/IP protocol. In this way, each module can be written in different programming languages (as long as the communication protocol is satisfied); further more, each module can be installed on a specific computer that is connected, together with the other computers supporting the other running modules, to a network (internet or intranet).

The BCI2000 system does not impose any restriction regarding the number of the acquisition channels, the sampling frequency and the complexity of the processing algorithms. Moreover, it is offering the possibility of the *offline* analysis by means of storing the acquired data.

The documentation of the system and the sources for each block are freely available and they can be downloaded from a *web* site [4] only for educational and research applications. Further more, several already implemented algorithms are offered. Until now, due to the possibility to freely access the source code, approximately 120 laboratories all around the world adopted this system.

I. The System Requirements

Having in view all the above mentioned researches, the large diversity of data processing and classification techniques used within the framework of the actual BCI systems, we propose in this paper the **implementation of a development platform for BCI applications based on a new concept.** The new platform will have the following fundamental characteristics:

- **autonomy**, given by the possibility of the system to be mounted on a mobile device (e.g. on a wheel chair controlled by a completely paralyzed subject) – in this case, restrictions regarding the system's power consumption and the computational costs have to be taken into consideration;
- ability to **process in real time** the acquired signals;
- **flexibility and independent operating mode** – due to the large variability of existing cerebral affections, of the phenomenon whished to be analyzed, the system has to be able to be configured to work with different types of methods and algorithms, combined in different structures and without the modification of the other parts of the system that are not directly involved;
- **scalability** – with no restrictions on the magnitude of the data sets;
- **ability to *offline* analyze the data sets.**

II. The System Implementation

The system is composed of the following main parts:

1. A Mindset 24 EEG acquisition system. The Mindset is a biomedical device able to acquire the brain electric activity. It is connected to the PC104+ embedded system through a SCSI interface and it is able to acquire the EEG signal using 24 independent channels, with a 16 bits analog to digital converter on each acquisition channel. This particular system was chosen mainly because it is based on open software architecture.
2. The embedded PC is a PC104+ system produced by Kontron company. The PC104+ is a MOPSlcd7 system having an Intel Pentium III processor at 500 MHz, 256 MB of RAM, 1 GB of USB flash hard disk, a 10/100 BaseTx Ethernet incorporated network adapter, an USB interface and an embedded BIOS.
3. A software platform that complies with all the previously presented requirements. The schematic of the software application and the data flow are presented in Fig. 1.

The new introduced concept that regards this BCI development platform is mainly based on the **dynamic modularity of the system**. In this way, each module can be independently modified; in addition, by sending a new module over the communication port, the new module automatically replaces the old one. By dynamically loading the module, the re-initialization of the entire application becomes unnecessary and, thus, the system will transparently use the new module.

Consequently, only by understanding the manner of the intermodule communication and without any knowledge of the whole system (architecture, other algorithms used in the rest of the system), a researcher will have the possibility to focus only on the development of a single algorithm embedded into a single module, disregarding the rest of the system. **In this way an abstraction of the global working principle of the system is obtained**.

The Dynamic Link Libraries are used to create all the user-defined modules: EEG acquisitions, data storages, signal preprocessing, feature extractions, classifications, post-processing and output command.

Since these DLLs can be developed in different languages (e.g. Visual C++, LabWindos CVI, Borland C++ Builder, Visual Basic, Borland Delphi etc.), the **independence of the system is not one related only to that of the modules** (of their particular combination in order to solve a problem), **but also to its independence with respect to the development environments used in the design of the modules**.

All the DLLs are dynamically loaded (explicitly link) by the frame level of the application, Fig. 2. Based on the Windows SDK functions, LoadLibrary and FreeLibrary, each DLL module can be independently loaded or discarded in respect to the other DLLs and without restarting the whole BCI system.

The data transfer between the modules is generally based on data arrays – mainly vectors. The modules consist of simple functions that receive the addresses of the arrays, as well as, the information regarding their dimensions (number of lines and columns); further, the modules return other data arrays, together with their dimensions. In particular, in order to store and to represent the data we chose the vector technique, given its simplicity of data allocation and manipulation. Even if the EEG data acquisition system acquires two dimensional arrays (e.g. 8 channels and 32768 samples on each channel), the storage and the access to a particular sample of the vector is very simple – *dataBuffer* [*i* + *j* **channelLengh*], where *channelLengh* = 32768, *j* is the channel number and *i* is the sample number.

The proposed system was first developed using a standard PC. Nevertheless, in all the development stages we had in mind that, at the end, the software will be executed by a PC104+ embedded system. The final tests were done on the embedded system.

The BCI applications are considered data-intensive and computing-intensive. Consequently, the frame level application of the system, Fig. 1, is built on the concept of being as less as possible expensive (both, data and computing). From this reason, the frame level application only loads/unloads the DLL modules, dynamically allocates the data arrays, sends from a function to another the addresses of vectors and their dimensions and, whenever it is required, it fires off the exported functions from the DLLs.

Each module is composed of a number of standard functions that must have standard names in all the modules, Table 1. These functions define the unique functionality of the modules.

In Table I the "****" symbols represent a specific name associated with each specific module – "Input" for the acquisition module, "PreProc" for the signal preprocessing module, "FeatureExt" for the features extraction block, "Classif" for the functions associated with the classification DLL, "PostProc" and "Output" for the modules used in information postprocessing and, respectively, system output.

It is not obligatory that all modules to include all the functions presented in Table 1, but only those specific functions necessary to implement their particular functionalities.

TABLE I.

THE STANDARD NAMES FOR THE EXPORTED FUNCTIONS

Exported function name	***Function task***
*exemplarStart_****	This function is executed before each feature vector is presented to the classification module.
*epochStart_****	This function is fired up before each epoch is started.
*moduleReset_****	The function is used when the frame level application resets the entire system.
*moduleInit_****	This function is fired up in order to initialize each particular module.
*modulePerform_****	The function implements the main functionality of the module
*moduleFree_****	Discards all data that were previously dynamically allocated in the DLL.
*moduleInfo_****	This function supplies different particular information regarding the state and the functionality of a module.

The moduleInit_**** function initializes the module and return to the frame level of the application an integer value that is directly correlated with the number and type of the internal implemented functions into the DLL. Based on these initializations given directly (as function arguments or supplied by a user through the one or several user interface panels) the functionality of the entire module is settled.

When a DLL is loaded into the virtual address space of the current process (the frame level of the application) the *moduleInit_**** is also executed; in this mode the DLL can initialize any instance of data and structures.

III. RESULTS

Up to now, in order to test the concepts of the proposed system, we implemented the following modules: signal preprocessing, features extractions, classification and the system output. We performed two types of tests: on-line tests and off-line tests.

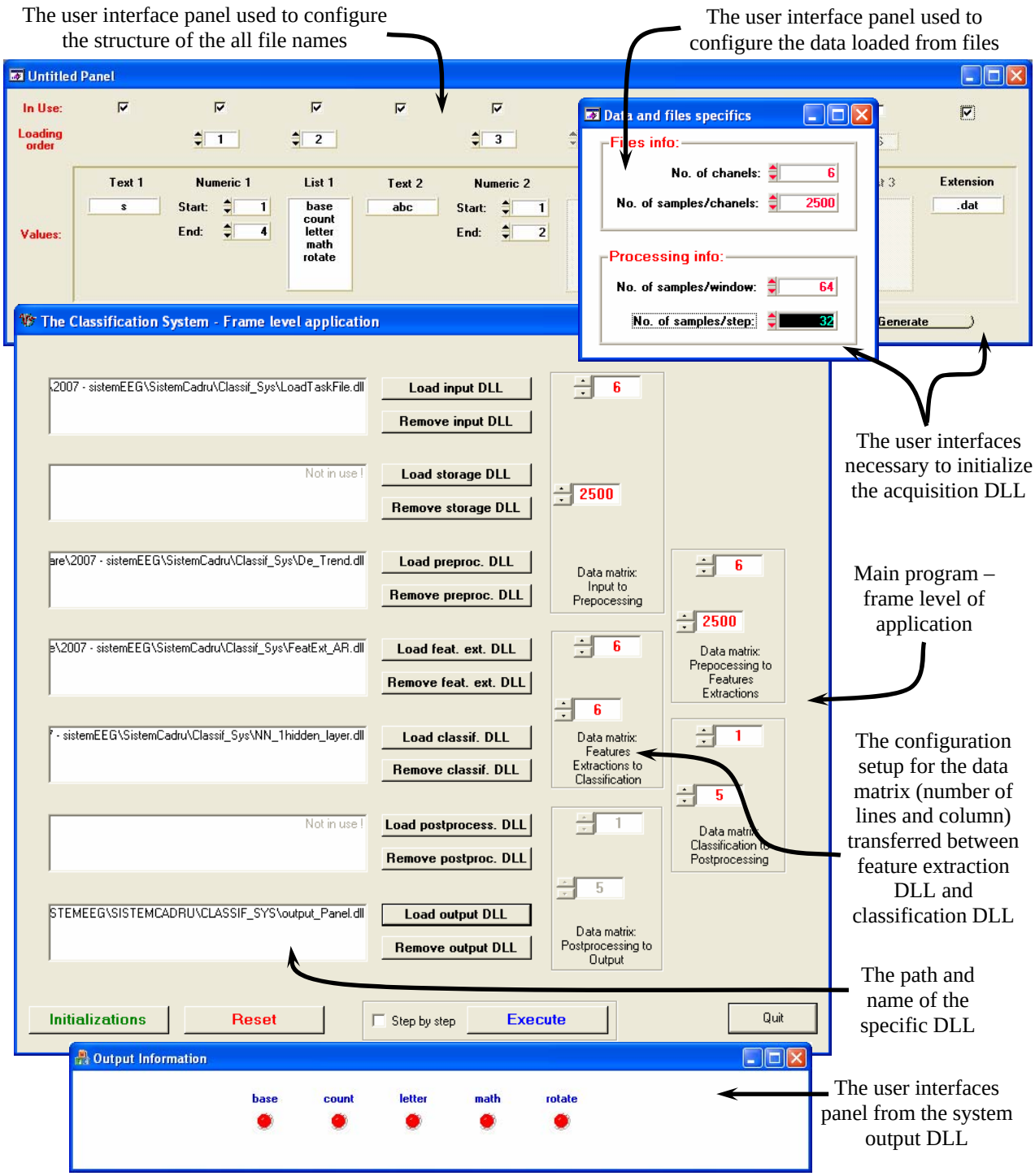

Figure 2. The user interface panels from the frame level application and from several DLLs in the initialization stage of the system

In the off-line tests the EEG acquisition system was only simulated using a DLL that loads files previously stored on the PC104+ flash hard disk. This last DLL represents another independent module used by the frame level applications, Fig 2.

For this test we used the same EEG database that was also used by us in two other previous studies [10], [11], and that was freely provided by the Colorado State University, Department of Computer Science [12]. In these EEG time series, the EEG signal was recorded simultaneously from 6 electrodes corresponding, in the International 10-20 system, to the C3, C4, P3, P4, O1 and O2 positions on the scalp. During these recordings the 4 involved subjects performed (without verbalization, with eyes closed) the following mental tasks:

basic cerebral activity (the subject was relaxed as much as possible), *writing* (mental composition of a letter), *mathematic calculus* (multiplication), *visual counting* (imagining a table on which sequential numbers were written) and *rotation of a geometric figure* (the visualization of a rotating three-dimensional object). All channels were referred to the right mastoid A2. They were digitally sampled at 250 Hz and each recording lasted 10 s. The system output module presents the classification result on the PC104+ monitor using five virtual LEDs associated with the tasks, Fig. 2.

In order to classify the tasks – having as final goal the aim to test the EEG bio-instrumental system and not to obtain better performances –, we used the same approach presented in [10]. For each channel 6 AR coefficients were extracted; thus, a feature vector of 36 elements was obtained for each consecutive time interval. In the classification of the feature vectors one hidden layer backpropagation trained neuronal network was used.

All the neural networks used on this research were developed, built and tested using the NeuroSolution software package. The C++ code used in the classification module was also generated with the same environment.

The classification results, obtained with our EEG bio-instrumental system and presented in Table II, are comparable with the ones presented in [10]. In both analyses the methods for preprocessing, features extractions and classification were the same and all the parameters used by these methods were identical.

Once again, **we want to stress that the main goal of this paper was only to test the concepts of the new proposed system and not to obtaining superior classification performances.**

TABLE II.
PERFORMANCES OBTAINED ON THE CROSS-VALIDATION SET

Result assigned / Real classes	baseline	count	letter	math	rotate
baseline	39.2	30.5	18.1	9.2	3.0
count	22.3	62.4	0	13.1	2.2
letter	5.2	6.1	55.3	19.3	14.1
math	4.2	5.3	14.1	75.0	1.4
rotate	17.3	14.2	6.1	6.1	56.3

The on-line obtained results were similar with the ones presented above.

The dynamic modularity of the system was also tested in two other main trials.

First, a suboptimal neural network was built, trained – in order to obtain the maximum of the permitted performances – and implemented into a dynamic link library. This neural network had the same dimensionality of the input feature space like the optimal neuronal structures presented above. Based on some very simple actions (like stopping the execution of the system, removing the existing classification system, loading the new one and starting the system) performed from the frame level user interface panel and without interacting with the other modules, Fig. 2, the behavior of the system was dynamically changed, without recompiling the entire application.

In a second test, a new neural network was implemented into a DLL, the only difference consisting in a larger input feature space. Eight auto-regressive coefficients were extracted from each channel. Loading the new neuronal classifier and changing the dimensionality parameters of data matrix (representing the data flow between the features extractions and the classification dynamic link library) – were the only steps required to dynamically implement the new structure. As a result, we obtained a different system having different characteristics.

As a conclusion, the EEG bio-instrumental system behaved excellent and proved its capability to interpret the cerebral activity of the brain and to transmit further commands.

IV. CONCLUSIONS

Conclusively, through the tests we done the new developed EEG bio-instrumental system demonstrated: its flexibility and independent operating mode, its ability to process in real time the EEG signals and, also, the simplicity of the idea that generates low data-intensive and computing-intensive burden at the frame level of the application.

Such an implementation of the system is capable to run on a PC104+ and on a PC system without any difficulty as long as the operating system is able to support the software application.

The flexibility of the system is also revealed by the unsophisticated implementation and by the straightforward usage of different modules, in any desired combination. All these capabilities facilitate the comparison of the results, thus helping in the best method selection process. Due to the simple communication algorithm, of call function type, one gets rid of the TCP/IP communication delays (that exist on a similar BCI2000 system); also, the understanding level of the concepts that underlie the communication is minim, thus facilitating the implementation method and increasing the system working speed.

In conclusion, this EEG bio-instrumental system shows to be a promising development platform for a large field of BCI applications.

ACKNOWLEDGMENT

The Grant 2735/19.05.2006, theme 26, code CNCSIS 17 of the National University Research Council has supported the research for this paper.

REFERENCES

[1] C. Guger, A. Schlögl, C. Neuper, D. Walterspacher, T. Strein, and G. Pfurtscheller, "Rapid prototyping of an EEG-based brain-computer interface (BCI)," IEEE Trans. on Neural Syst. and Rehab. Eng., vol. 9, no. 1, pp. 49-58, 2001.

[2] G. Schalk, D.J. McFarland, T. Hinterberger, N. Birbaumer, and J.R. Wolpaw, "BCI2000: a general-purpose brain-computer interface (BCI) system," IEEE Trans. on Biomed. Eng., vol. 51, no. 6, pp. 1034-1043, 2004.

[3] S.G. Mason, and G.E. Birch, “A general framework for brain-computer interface design”, IEEE Trans. on Neural Sys. and Rehab. Eng., vol. 11, no. 1, pp. 70-85, 2003.
[4] The BCI2000 system web address for doccumentation and sources code, http://www.bci2000.org/BCI2000/Home.html.
[5] G. Schalk, D.J. McFarland, T. Hinterberger, N. Birbaumer, and J.R. Wolpaw, “BCI2000: a general-purpose brain-computer interface (BCI) system,” IEEE Trans. on Biomed. Eng., vol. 51, no. 6, pp. 1034-1043, 2004.
[6] J.R. Wolpawa, N. Birbaumerc, D.J. McFarlanda, G. Pfurtschellere, and T.M. Vaughan, “Brain-computer interfaces for communication and control,” Clin. Neurophys., vol. 113, pp. 767-791, 2002.
[7] H. Serby, E. Yom-Tov, and G.F. Inbar, “An improved P300-based brain-computer interface”, IEEE Trans. on Neural Sys. and Rehab. Eng., vol. 13, no. 1, pp. 89-98, 2005.
[8] C. Guger, A. Schlögl, C. Neuper, D. Walterspacher, T. Strein, G. Pfurtscheller, “Rapid prototyping of an EEG-based brain-computer interface (BCI),” IEEE Trans. on Neural Syst. and Rehab. Eng., vol. 9, no. 1, pp. 49-58, 2001.
[9] J.M. Heasman, T.R.D. Scott, L. Kirkup, R.Y. Flynn, V.A. Vare, and C.R. Gschwind, “Control of a hand grasp neuroprosthesis using an electroencephalogram-triggered switch: demonstration of improvements in performance using wavepacket analysis,” Med. & Biol. Eng. & Comp., vol. 40, pp. 588-593, 2002
[10] V.A. Maiorescu, M.C. Serban, and A.M. Lazar, “Classification of EEG signals represented by AR models for cognitive tasks - a Neural Network Based Method”, in Proc. of Int. Symp. on Signal Circ. and Sys., Iasi, Romania, July 10-11, vol. 2, pp. 441-444, 2003
[11] M.C. Serban, and D.M. Dobrea, “A cognitive tasks’ discrimination study,” in Proc. of the Inter. Sym. on Sig. Circ. and Sys., Iasi, Romania, July 14-15, 2005, vol. 2, pp. 805-808.
[12] EEG data base, http://www.cs.colostate.edu/~anderson/, 2000.