Embedded-BCI: assessment of parallelizing computations on an embedded system

Amin Zammouri, Abdelaziz Ait Moussa, Sylvain Chevallier

Embedded-BCI: assessment of parallelizing computations on an embedded system

Amin Zammouri[1,2,*], **Abdelaziz Ait Moussa**[1], **Sylvain Chevallier**[2]

[1] Department of Computer Science, Faculty of Sciences, Mohammed First University, 60000 Oujda, Morocco
[2] Laboratoire d'Ingénierie des Systèmes de Versailles, Université de Versailles Saint-Quentin-en-Yvelines, 78140, Vélizy, France
a.zammouri@ump.ac.ma, a_aitmoussa@yahoo.fr, sylvain.chevallier@uvsq.fr

Abstract: Using Brain-Computer Interfaces (BCI) as an assistive technology aims at providing an innovative solution adapted to subjects' disabilities. BCI either provide a new interface for controlling solution mobility (e.g. wheelchair) or monitoring the state of user during his/her journey. This would be possible by implementing these interfaces on Embedded Systems (ES). However, because of the BCI sophisticated data processing and the ES limited computation performances, the computation time for a real-time use of the BCI on an ES is a limitation. Hence in this work, we investigate and evaluate the parallelization and acceleration performances, on a Raspberry Pi 2 model B (RPi) board, of an STFT-based algorithm for estimating cognitive workload from an Electroencephalographic (EEG) signal. This is done based on multi-core CPU and GPU architectures of the used RPi. Results show that the parallelized implementation using the CPU runs up to $\times 4$ faster than a simple implementation. Compared to CPU of intel-CORE i3 processor, the GPU of the RPi revealed large difference in computation time.

Keywords: BCI, embedded system, parallelization, CPU, GPU, multicore.

I. INTRODUCTION

Nowadays, embedded systems become the highlights of the artificial intelligence (AI) progress. These electronic and computer systems are designed to perform specific tasks while ensuring a high autonomy. This autonomy lies in their capacities to manage their available resources for interacting with all other systems comprising their environment. Although the use of these systems, in recent years, was limited to areas of military, currently the deployment of such systems is widely applied in many other fields, especially transport. Beyond this, ES integrate more and more the human everyday life by seeking to design new assistance tools for disabled people or in situations of extreme dependence. A crucial aspect is to take into account the specificities of each individual and propose technical solutions adapted to their residual motor capacities [1]. In this aspect which is sought to minimize decision errors, the development process is particularly harsh and hence involving more and more formal techniques.

Mobility represents a basic need for people with motor deficiencies to integrate and participate in the social everyday life [2]. In this context, the wheelchair is the most commonly used assistive device to allow both internal and outside mobility. In often cases this device is controlled manually with a joystick. In the case where no residual motor ability is available, paradigms based on non-muscular flows can be used (e.g. eye movements, galvanic skin response and heart rate variation). However, regarding the variability of disabled users, the use of brain signals may represent the reliable control mean. In the context of assistive mobility, Brain-Computer Interfaces (BCIs) provide systems of direct control through non-muscular channels based on brain signals [3],[4],[5]. In this realm, Rebsamen in [6] introduces a control strategy for wheelchair driving based on the P300 Event-Related Potential (ERP) component. Through a GUI, the BCI-based wheelchair system proposes to the user to select his/her decisions to move the wheelchair in a typical building. The paradigm used in this strategy exploits the generation of a positive deflection, P300, in the brain signal. This deflection is measured in the central brain area 300 milliseconds after the reception of a stimulus. This paradigm was introduced for the first time by Farwell and Donchine in their P300 Speller [7]. In the same idea of developing technical solutions to assist mobility and increase motor capacities of disabled people, Tonin in [8] presents a BCI-based approach for controlling a telepresence robot by users with disabilities. Based on their spontaneous brain activity, users drove a telepresence robot from their clinic more than 100 Km away. In order to facilitate the navigation, this approach combines concepts of motor imagery and shared control. The developed system makes use of imagination of movements (e.g. imagining the movement of hands) and an obstacle detector for safety and helping users to keep full control on the robot driving. Results from this approach show that the incorporation of shared control reduces users' mental efforts.

A part from using BCI for controlling robotic devices [9],[10], [11], recent works were focused on assessing and evaluating mental efforts and workload while using mobility devices. In [12] authors developed an EEG-based approach in order to assess and monitor changes in drivers' cognitive states while using a virtual reality

(VR)-based driving environment. This unsupervised approach generates, in every driving session, a statistical model of the alert state of the driver. Authors demonstrated that the deviation of drivers' cognitive state from the alert model covaries with the driving performance. This approach, and many others, takes advantage of the spectral aspect of the brain electrical signal. It consists in computing the EEG power spectrum using a Short Time Fourier Transform (STFT). The STFT allows to calculate the relative energy into the different EEG bands [13], namely, Delta [0.5-3 Hz] (δ), Theta [4-7 Hz] (θ), Alpha [8-11 Hz] (α), Beta [12-30 Hz] (β) and Gamma [>30 Hz] (γ).

Despite the innovative aspect provided by BCI systems, their uses in the everyday life remain limited and suffer from several constraints. These constraints are due to the large number of sophisticated processings required for artefacts removal, calibration sessions and classifications. On another hand, incorporating BCI on ES is a promising perspective in the context of mobility assistance for people with limited motor capacities. But to achieve designing such solution while ensuring a reliable rate for use in everyday life, several factors must be taken into account carefully. The rigorous processings such as Independent Component Analysis (ICA) [14], Principal Component Analysis (PCA) [15] and regression [16] involves using ES with high computing performances (in term of memory). This constraint immediately implies the constraint of the solution cost.

In this work we evaluate the usability of a low cost ES to design future BCI applications. Our investigation aims at studying, on a Raspberry Pi 2 B (RPi) board, the performances of parallelizing an STFT-based algorithm to estimate the subject's cognitive workload from EEG recordings. The approach we propose overcomes the problem of weak computing performances that can make limitations of a real-time use of BCI on the RPi. Our idea consists in parallelizing the STFT-based algorithm computations based on multi-core, Central Processing Unit (CPU) and Graphical Processing Unit (GPU) architectures of the RPi. We focused our study on the STFT parallelizing since it is the most time-consuming part of the brain workload estimation process from a multi-channel EEG signal. This work exploits two BCI experimental protocols used in our previous works. The first protocol is based on a system which integrates BCI as a complementary communication channel [1]. This hybrid system integrates an SSVEP-based BCI and a 3D touchless interface based on IR-sensors [17]. In the second protocol we couple a passive BCI to an Intelligent Tutoring System (ITS). Experimental results of our investigation show that the parallelized computations on the used RPi board ensure high performance in terms of time computation in a real-time use.

The paper is organized as follows. In section 2 we detail the Fourier transform algorithm. Section 3 presents our methodology for parallelization of the STFT algorithm. Experimental results are described and discussed in sections 4. Finally, conclusions are drawn in the last section.

II. FOURIER TRANSFORM ALGORITHM

Fourier transform is a widely used method for studying non-stationary signals as it gives the time-frequency distribution for many signals:

$$S(f) = \int_{-\infty}^{+\infty} s(t)e^{-j2\pi ft}dt \tag{1}$$

by approaching the integral by a sum of rectangular areas of time T_e and by limiting the integration time to the interval $[0, (N-1) * T_e]$ we obtain:

$$S(f) \approx T_e \sum_{n=0}^{N-1} s(nT_e)e^{-j2\pi fnT_e} \tag{2}$$

which gives for frequency values $f_k = k\frac{f_e}{N}$:

$$\begin{aligned} S(f_k) &\approx T_e \sum_{n=0}^{N-1} s(nT_e)e^{-j2\pi \frac{nk}{N} f_e T_e} \\ &\approx T_e \sum_{n=0}^{N-1} s(nT_e)e^{-j2\pi \frac{nk}{N}} \end{aligned} \tag{3}$$

It is not a sophisticated approximation of $S(f)$ but is used in practice, as Discrete Fourier Transform (DFT), since there is an effective computing algorithm known as Fast Fourier Transform (FFT) [18] . The computation of DFT requires N^2 complex operations while using the FFT needs only $Nlog_2N$.

III. METHODS AND MATERIALS

A. Experimental setups

The first experimental protocol involves controlling an exoskeleton arm using a touchless interface with 5 IR-sensors which allow the displacements in different spatial positions. The touchless interface is complemented by an SSVEP-based BCI. The experimentations rely on the ESTA robotic exoskeleton [19]. The EEG data acquisition system uses the g.Mobilab+ device using an acquisition frequency of 256 Hz. The acquisition device is based on 8 channels placed according to the 10-20 international system. All electrodes refer to the right or left earlobe and the ground is placed on Fz as depicted in Figure 1. For SSVEP stimulations, a microcontroller is set up to flash stimuli light emitting diodes (LED) at frequencies 13 Hz, 17 Hz and 21 Hz.

The second experimental protocol consists in measuring the brain activity during the use of an ITS. Equipped with a control tool, the user navigates on the ITS. The BCI-based ITS adapts and presents the learning content to the user depending on his/her mental state and effort. EEG data are measured continuously basing on the g.Mobilab+

acquisition device with 8 channels (Oz, O1, O2, POz, PO3, PO4, Pz, Iz) and using a sampling rate of 256 Hz. The EEG data are filtered using a pass-band filter for 1-30 Hz.

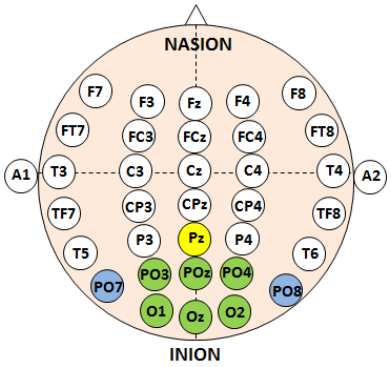


Figure 1. Placement of used electrodes. Green color: electrodes used in both the two experimental setups. Blue color: electrodes used in the ESTA-based experimentation. Yellow color: electrodes used in the ITS-based experimentation.

On another hand, our study exploit a model 2 B RPi board with 1 GB of RAM. The system on chip RAM (Broadcom BCM2836) of the used RRi board contains both a CPU and GPU with independent performances to each other. We decided to use this RPi board model given the multicore architecture that it incorporates.

B. CPU implementation

Our parallelization approach on the CPU of the RPi is straight-forward. The implementation makes use of the multi-cores of the RPi. Thanks to the multi-threading architecture incorporated in the used RPi, The 8 EEG channels are distributed among N_{th} threads. On another hand, the implementation takes advantage of the OpenMP library in order to automatically distribute the execution of the STFT code on the used threads. Thus, each thread executes the code separately and calculates its own STFT. An overview of this process is presented in Figure 2.

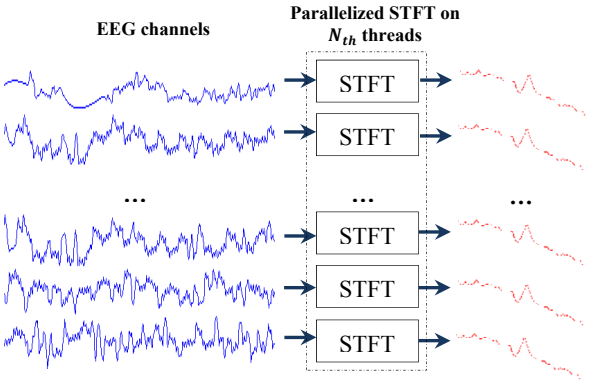


Figure 2. Parallelization scheme on the CPU of RPi

C. GPU implementation

To our knowledge, there is no library or programming language, like CUDA or OpenCL, intended particularly for the parallelization on the GPU of RPi. Works from literature, and which exploited the RPi GPU, used optimized assembler codes. The RPi board has 12 cores inside its GPU, each known as QPU (for Quad Processing Unit). The STFT algorithm implementation could be optimized leveraging these QPUs. A QPU is a 16 Single Instruction Multiple Data (SIMD) processor which allows disposal of vectors of 16 values. Therefore an execution using vectors of 16 values can be done in parallel. On another hand, as the STFT computation uses complex numbers, this implementation requires only two registers to store 16 real number and 16 imaginary number. We denote by *Step* the computing time unit in which a QPU consumes 16 points. This implies that a QPU performs two steps (one for reals and another for imaginaries) when consuming the 16 points. Thus computing the STFT on 128 points requires $N_{QPU} = 128/16 = 8$ QPUs. The computing time unit in which the 8 QPU consume the 128 points in parallel is denoted by *Pass*. Thus for computing the STFT on 256 points, 8 QPUs are required using two passes. At the end of each pass, each QPU accesses the Vertex Pipeline Memory (VPM) of the RPi memory in order to write its outcomes.

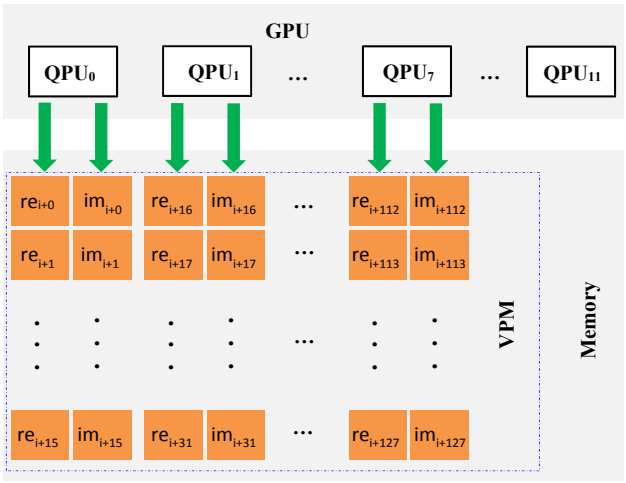


Figure 3. Parallelization scheme on the GPU of RPi.

IV. RESULTS AND DISCUSSION

Here we present results with particular attention on computation time. CPU implementation revealed reliable performances. The STFT algorithm was computed using signal windows with various lengths. We considered window lengths of $1s$, $2s$ and $4s$. Using EEG data of our experiments, these windows lengths respectively correspond to points numbers of 256, 512 and 1024. Figure 4 shows the gain in the computation time (per seconds) for these different window lengths while using N_{th} threads. There are 3 different parts on the chart. For 1 to 4 threads, the gain in calculation time is very close from the theoretical gain which reflects excellent parallelization. All threads are processed by the processor cores. The system performs as a quad processor. Between 5 and 7 threads, the curve does not follow the theoretical curve and the gain in computation time decreases.

Threads management takes too much time and slows the program. This is due to the fact that the number of threads is less than and not proportional to the number of cores. Beyond 8 threads the gain increases and remains the same even when using 12 threads. On another hand, results show that the gain in computation time increases when increasing the number of points on which the STFT is computed. This reflects that the parallelization is effective when working with window lengths larger than $4s$.

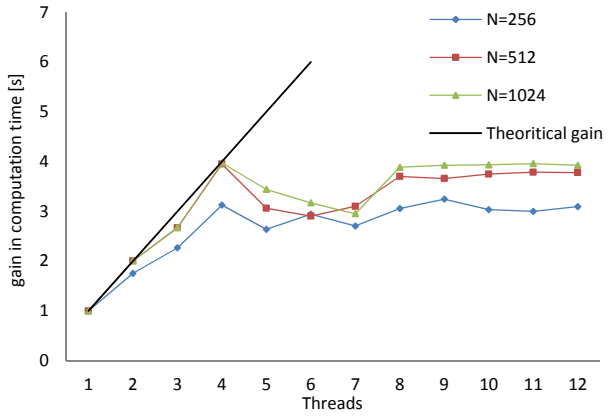
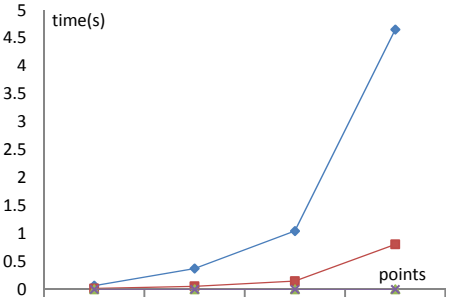


Figure 4. Comparison of gain in computation time while using different threads.

A part from this, we measured the performance of the GPU implementation using 900 MB of the RPi memory. The implementation was tested with 5 different window lengths. Figure 5 shows resulting computation times from CPU of the RPi, CPU of intel-CORE i3 processor and GPU of the RPi. The x-axis reports the different lengths on which the STFT was computed. The y-axis indicates computation time of optimized implementations. Results illustrate important computational performances when using the RPi GPU. The execution time difference between the RPi CPU and GPU implementations is due to the effective access to the VPM using an assembler code.



	256	512	1024	2048
CPU RPi	0.065	0.372	1.044	4.652
CPU i3	0.0147	0.0531	0.147	0.805
GPU RPi	3,000E-08	4,300E-08	6,200E-08	001E-04
GPU i7	2,627E-08	2,629E-08	3,447E-08	3,448E-08

Figure 5. Comparison of computation time (per seconds) of optimized implementations of the STFT algorithm on CPU of RPi, CPU of intel-CORE i3, GPU of RPi, and the GeForce Titan GPU of intel-CORE i7.

Comparing performances of the RPi GPU to those of the CPU of intel-CORE i3 processor revealed the huge gain in computation time offered by the RPi GPU when computing the STFT on larger time-windows.

However the RPi GPU implementation offers promising optimization performances, its use in the design of innovative embedded solutions remains limited. This limitation is due to absence of libraries or programming languages, such as CUDA or OpenCL, which are designed specifically for parallelization on the RPi GPU. Therefore, at present, the development of ES that leverages the GPU of the RPi imposes very high skills in low-level programming using assembler. However, through findings presented in this work, we demonstrate that the parallelism on the CPU of the RPi using the OpenMP library is a robust and efficient alternative solution for accelerating real-time processing of data from time-windows of lengths between 1 and 2 seconds.

CONCLUSION

In this work we studied and assessed performances of parallelization and acceleration of computation on the RPi board for a reliable use of BCI applications. Our investigation consisted on parallelizing a STFT-based algorithm for estimating the brain workload. Our study focused on the STFT parallelizing since it is the most time-consuming part of the brain workload estimation process from a multi-channel EEG signal. Based on the multi-threading and multi-cores architectures included in the RPi, the optimized CPU implementation runs up to $\times 4$ faster. On another hand, based on assembler implementation, we evaluated computation performances of parallelizing the STFT algorithm on the RPi GPU. Compared to CPU of intel-CORE i3 processor the GPU of the RPi revealed large difference in computation time. The whole results are motivating to design future RPi-based mobile systems which take advantage from BCIs innovative aspect (e.g. autonomous mobile robots).

REFERENCES

[1] E. K. Kalunga, S. Chevallier, O. Rabreau, and E. Monacelli. “Hybrid interface: Integrating BCI in multimodal human-machine interfaces,” in *Proc. AIM*, 2014, pp. 530–535.

[2] R. Rupp, S. C. Kleih, R. Leeb, J. del R. Millan, A. Kübler, and G. R. Müller-Putz. “Brain–Computer Interfaces and Assistive Technology,” in *Brain-Computer-Interfaces in their ethical, social and cultural contexts*, vol. 12, G. Grübler and E. Hildt, Eds. Dordrecht: Springer Netherlands, 2014, pp. 7–38.

[3] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan. “Brain-computer interfaces for communication and control.” *Clin. Neurophysiol.*, vol. 113, no. 6, pp. 767–791, Jun. 2002.

[4] G. Pfurtscheller, D. Flotzinger, and J. Kalcher. “Brain-Computer Interface—a new communication device for handicapped persons.” *J. Microcomput. Appl.*, vol. 16, no. 3, pp. 293–299, Jul. 1993.

[5] T. Castermans, M. Duvinage, G. Cheron, and T. Dutoit. “Towards Effective Non-Invasive Brain-Computer Interfaces Dedicated to Gait Rehabilitation Systems.” *Brain Sci.*, vol. 4, no. 1, pp. 1–48, Dec. 2013.

[6] B. Rebsamen, E. Burdet, C. Guan, C. L. Teo, Q. Zeng, M. Ang, and C. Laugier. "Controlling a wheelchair using a BCI with low information transfer rate," in *Proc. ICORR*, 2007, pp. 1003–1008.

[7] L. A. Farwell and E. Donchin. "Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials." *Electroencephalogr. Clin. Neurophysiol.*, vol. 70, no. 6, pp. 510–523, Dec. 1988.

[8] L. Tonin, T. Carlson, R. Leeb, and J. del R Millan. "Brain-controlled telepresence robot by motor-disabled people," in *Proc. EMBC* 2011, pp. 4227–4230.

[9] C. J. Bell, P. Shenoy, R. Chalodhorn, and R. P. N. Rao. "Control of a humanoid robot by a noninvasive brain–computer interface in humans." *J. Neural Eng.*, vol. 5, no. 2, pp. 214–220, Jun. 2008.

[10] R. Tomari, R. R. A. Hassan, W. N. W. Zakaria, and R. Ngadengon. "Analysis of Optimal Brainwave Concentration Model for Wheelchair Input Interface," *Procedia Comput. Sci.*, vol. 76, pp. 336–341, 2015.

[11] G. Cisotto, S. Pupolin, M. Cavinato, and F. Piccione. "An EEG-Based BCI Platform to Improve Arm Reaching Ability of Chronic Stroke Patients by Means of an Operant Learning Training with a Contingent Force Feedback." *Int. J. E-Health Med. Commun.*, vol. 5, no. 1, pp. 114–134, 2014.

[12] N. R. Pal, C.-Y. Chuang, L.-W. Ko, C.-F. Chao, T.-P. Jung, S.-F. Liang, and C.-T. Lin. "EEG-Based Subject- and Session-independent Drowsiness Detection: An Unsupervised Approach." *EURASIP J. Adv. Signal Process.*, vol. 2008, no. 1, pp. 519-480, 2008.

[13] A. Picot, S. Charbonnier, and A. Caplier. "On-line automatic detection of driver drowsiness using a single electroencephalographic channel," in *Proc. EMBC* 2008, pp. 3864–3867.

[14] W. Zhou and J. Gotman. "Automatic removal of eye movement artifacts from the EEG using ICA and the dipole model." *Prog. Nat. Sci.*, vol. 19, no. 9, pp. 1165–1170, Sep. 2009.

[15] O. G. Lins, D. T. W. Picton, P. Berg, and M. Scherg. "Ocular artifacts in recording EEGs and event-related potentials II: Source dipoles and source components." *Brain Topogr.*, vol. 6, no. 1, pp. 65–78, Sep. 1993.

[16] A. Schlögl, C. Keinrath, D. Zimmermann, R. Scherer, R. Leeb, and G. Pfurtscheller. "A fully automated correction method of EOG artifacts in EEG recordings." *Clin. Neurophysiol.*, vol. 118, no. 1, pp. 98–104, Jan. 2007.

[17] H. Martin, S. Chevallier, and E. Monacelli. "Fast calibration of hand movement-based interface for arm exoskeleton control," in *Proc. ESANN*, 2012, pp. 573–578.

[18] P. Duhamel and M. Vetterli. "Fast fourier transforms: A tutorial review and a state of the art." *Signal Process.*, vol. 19, no. 4, pp. 259–299, Apr. 1990.

[19] M. Baklouti, P.-A. Guyot, E. Monacelli, and S. Couvet. "Force controlled upper-limb powered exoskeleton for rehabilitation," in *Proc. IROS*, 2008, pp. 4202–4202.