Recreational Applications of OpenViBE: Brain Invaders and Use-the-Force

Anton Andreev, Alexandre Barachant, Fabien Lotte, Marco Congedo

Recreational Applications of OpenViBE: Brain Invaders and Use-the-Force

Anton Andreev, Alexandre Barachant, Fabien Lotte, Marco Congedo

CHAPTER 14: BRAIN-COMPUTER INTERFACES 2: TECHNOLOGY AND APPLICATIONS, EDITED BY MAUREEN CLERC, LAURENT BOUGRAIN, FABIEN LOTTE (WILEY-ISTE), 2016

Introduction

This chapter aims at providing the reader with two examples of open-source BCI-games that work with the OpenViBE platform. These two games are "Brain Invaders" and "Use-The-Force!" and are representative examples of two types of BCI: ERP-based BCI and oscillatory activity-based BCI. This chapter presents the principle, design and evaluation of these games, as well as how they are implemented in practice within OpenViBE. This aims at providing the interested readers with a practical basis to design their own BCI-based games. These two games are described hereafter.

Chapter written by Anton ANDREEV[1], Alexandre Barachant, Fabien LOTTE[2,] Marco CONGEDO*[1]
[1] GIPSA-lab, CNRS and Grenoble University, FRANCE
[2] Inria Bordeaux Sud-Ouest, FRANCE
*Corresponding Author
Marco.Congedo@gmail.com
GIPSA-lab, 11 rue des Mathématiques, Domaine universitaire - BP 46 - 38402, Grenoble, France.
tel: +33 (0)4 76 82 62 52
fax: +33 (0)4 76 57 47 90

The Brain Invaders

A P300-based Brain Computer Interface (BCI) enables the user to successively select symbols among an available set, without relying on any motor command. The symbols can be of any kind, such as alphanumeric characters (e.g., for spelling) or icons (e.g., the elements of a menu in a computer application). These BCIs exploit the well-known oddball paradigm, in which an infrequent task-related item (the target symbol) elicits a P300 Event-Related Potential (ERP) [WOL 2011]. By flashing symbols exhaustively, either one-by-one or in groups, it is possible to estimate the probability of each symbol being the one selected by the user. This is achieved evaluating the P300 elicited by each symbol once it has flashed. The complete set of flashes must be repeated a number of times to obtain reliable ERP estimations by means of trial averaging. The distinctive advantages of P300-based BCI are that the alphabet (the set of all available symbols) can be large (hundreds of symbols) and that 100% accuracy can be in principle obtained when allowing a sufficient number of repetitions. That is to say, with P300-based BCIs there is a direct trade-off between accuracy and speed of symbol selection. In the context of this chapter the low transfer rate is not considered a limitation, rather a challenge for the player, along the line of the reasoning in [NIJ 2009]. Nonetheless, we aim at video games progressing with a sustained pace. For this reason we have implemented several improvements over the basic P300 BCI paradigm [CON 2011].

The Brain Invaders is inspired from the famous vintage game *Space Invaders*. As most old-fashion video game the Brain Invaders proceeds by levels. To finish a level the user has to destroy a target alien, chosen at random within a grid of 36 aliens and which is indicated by a red circle at the beginning of the level. Aliens may be of different color. The target alien is always red. Aliens move with patterns that are specific to each level. A repetition of flashes consists in 12 flashes of groups of 6 aliens chosen in such a way that after each repetition each alien has flashed exactly two times. After each repetition the system assigns to each alien the probability of being the target according to the signal processing and classification method implemented in the OpenViBE platform and destroys the alien with the highest probability. If this alien is the target the level ends, otherwise this alien is eliminated and another repetition of flashes starts. The process is continued until the target alien is destroyed or until eight non-target aliens have been destroyed, after which another level starts. The current number of attempts per level is indicated by coloring the bullets on the bottom of the screen. During the game the cumulative score is shown to the player. The points obtained at each level are inversely proportional to the number of repetitions necessary to destroy the target. Figure 1 a) shows the welcome screen, (b) shows the simplest level, in which the aliens move altogether from the left to the right of the screen as in the original game Space Invaders, (c) and (d) show more complex levels, where aliens move according to elaborated patterns and several distracting aliens are colored green or red, like the target. The flashing time

is fixed and should be set in between 60 ms and 150 ms The inter-stimulus interval (ISI) is randomly drawn from an exponential distribution with mean 100 ms and bounded in the range [20…500] ms by drawing a random number until it falls in this range. The destruction is almost instantaneous after the last flash. Then a 2-sec break is allowed to relax and move freely, after which the new level starts. One game session is composed of 12 levels.

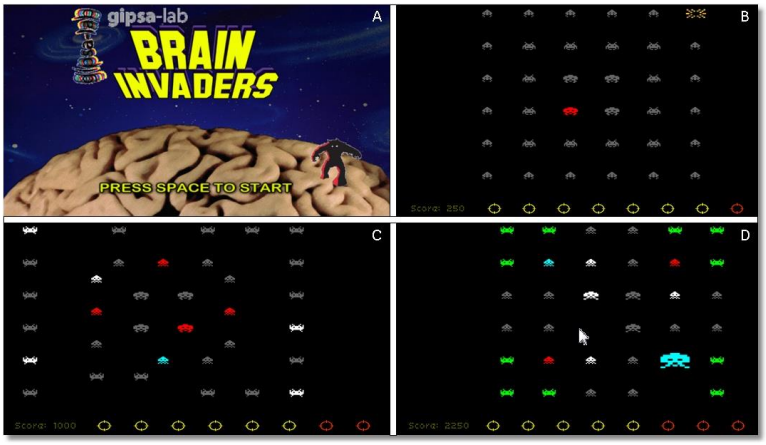

Figure 1: *Screenshots of the Brain Invaders user interface. See text for details*

Results

We present several results issued from an extensive evaluation of the Brain Invaders performed at GIPSA-lab in Grenoble. 24 subjects performed one session of the Brain Invaders. Seven of these subjects performed seven more sessions, twice a week, for a total of eight sessions. Each session consisted of two runs of the Brain Invaders, one using the typical training-test procedure (non-adaptive mode) and the other without any training using an initialization and an adaptation scheme (adaptive mode). In the non-adaptive mode the BCI is trained on a training session and the training is used to calibrate the classifiers to be used in the test session. In the adaptive mode the BCI is initialized with a training obtained on a user database and then continuously learn from the subject while the subject is playing. The two runs looked exactly identical to the subjects, in that in both cases a training session preceded a test session, however the training session was not used for calibration in the adaptive mode. The order of the two runs was randomized and the design was double-blinded; at any time neither the subject nor the experimenter could know in what mode the BCI was running. Data was acquired with a Porti amplifier (TMSi, The Netherlands) using 16 electrodes positioned at Fp1, Fp2, Afz, F5, F6, T7, Cz, T8, P7, P3, Pz, P4, P8, O1, Oz, O2, referenced by the amplifier to an hardware

common average, using a cephalic ground and sampled at 512 Hz. In online operation and for offline analysis EEG data were band-pass filtered in the range 1-20 Hz and downsampled to 128 Hz.

We present both some online results and offline results, the latter in order to compare the Riemannian minimum distance to mean (MDM) classifier [CON 2013], which is used by the Brain Invaders, with two popular state of the art algorithms [LOT 2007]: XDAWN [RIV 2011] and the stepwise linear discriminant analysis (SWLDA) [FAR 1988]. For XDAWN the two most discriminant spatial filters were retained. EEG data was then spatially filtered, decimated to 32 Hz and vectorized so as to classify the obtained 32x2 features with a regularized linear discriminant analysis (LDA), using an automatic setting of the regularization parameter [LED 2004], [VID 2009]. For the SWLDA, EEG data were decimated to 32 Hz and vectorized so as to feed the classifier with the obtained 32x16 features.

We begin by presenting several offline results of the performance pertaining to the non-adaptive mode, including the classic training-test setting and the cross-subject and cross-session initialization comparing several classifiers. We also present the online results obtained in the adaptive and non-adaptive mode. These latter results are the most relevant as they report the actual performance achieved by the Riemannian MDM algorithm in real operation. All performance results for this experiment are reported in terms of AUC (area under the curve).

Offline results: the "classic" training-test mode.

Fig. 2 shows the grand average (7 subjects x 8 sessions) AUC accuracy criterion for the three classification methods, obtained training the classifiers on the training run and testing on the test run ("Classic" column). Paired t-tests revealed that the mean AUC obtained by the MDM is significantly superior to the mean AUC obtained by the SWLDA method ($t_{(55)}$= 3.377, p=0.001), and equivalent to the mean AUC obtained by XDAWN.

Offline results: the cross-subject initialization.

These results are obtained using a leave-one-out method. Fig. 2 shows the grand average (7 subjects x 8 sessions) AUC accuracy criterion for the three classification methods obtained training the classifiers on the test data of all subjects excluding the one on which the performance are computed ("Cross-subject" column). As compared to the classic mode the average AUC with cross-subject transfer learning

is significantly lower for all classification methods ($p<0.002$ for all of them). This is an expected result as no information at all about the subject actually using the BCI is provided to the classifiers. Paired t-tests comparing the average performance of the three classification methods in the cross-subject mode reveal that the average AUC obtained by the MDM is marginally superior to the average AUC obtained by the SWLDA ($t_{(55)}$= 1.676, p=0.099) and by XDAWN ($t_{(55)}$= 1.755, p=0.085).

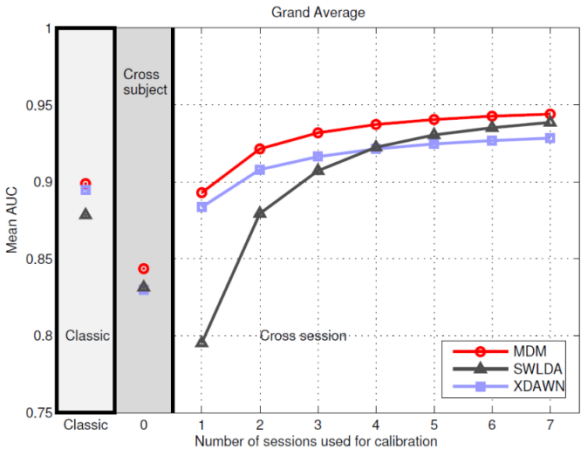

Figure 2: *Classic (training-test), cross-subject and cross-session offline AUC performance for the P300-based Brain Invaders BCI experiment. Results are the grand average of 7 subjects playing 8 sessions of the Brain Invaders. See text for details*

Offline results: the cross-session initialization.

These results are also shown in Fig. 2 ("Cross-session" column). The mean AUC is obtained initializing the classifier with any possible combination of *S* number of sessions among the eight available sessions and testing on the remaining 8-*S* sessions. The results are given for *S* in the range 1,…,7 and correspond to the average of all subjects and all combinations (which number depends on *S*). The MDM algorithm proves superior both in the rapidity of learning from previous subject's data and in the performance attained for all values of *S*, although for S=7 the performance of the SWLDA approaches the performance of MDM. Note that XDAWN, which is a spatial filter approach, performs fairly well even when only one session is available for training, but its performance grows slowly as more data is available for training. This is because the spatial filter is influenced negatively by the difference in electrode placements across sessions and, in general, by all factors that may change from one session to the other. On the other hand the SWLDA

classifier performs poorly when only one session is available for training, however it learn fast as the number of available sessions increase. This is because the SWLDA, being a “hard machine learning” approach, tends to perform well only when a lot of training data is available. So, XDAWN possesses fast learning capabilities, but lacks good transfer learning, whilst the opposite holds for SWLDA. The MDM algorithm possesses both desirable properties.

Online results: adaptation.

Finally, we show the actual online results for the adaptive and non-adaptive mode of functioning. Let us remind that the adaptive and non-adaptive runs were performed in a double-blinded fashion and randomized order. In online operation, starting from the second repetition the MDM uses the cumulated distance of all repetitions to select the alien with the highest probability. Hence, the number of repetitions needed to destroy the target (NRD) is a direct measure of performance: the lower the NRD the higher the performance. The generic classifier is calibrated using online data of the preceding sessions. The individual classifier is trained in a supervised way (the labels are known) during the experiment after each repetition. Of course, the current repetition (used to select the target) is added to the training set only after the classification output is used in order to avoid biasing the results. The weights of the initial classifier (generic) based on a database and the classifier training on-line on the subject while s/he is playing (individual) are set according to the current number of repetitions, that is, the individual classifier is weighted as alpha = min(1, Nrep/40) and the generic classifier as (1-alpha); in this way, the generic classifier is not used anymore after 40 repetitions. This value as been set arbitrarily based on pilot studies.

Figure 3 shows the mean and standard deviation NRD as a function of levels for the first session performed by all 24 subjects. As we can see, the non-adaptive MDM features a non-significant negative slope ($p=0.087$), meaning constant performance across levels, whereas the adaptive MDM features a significantly negative slope ($p=0.009$), meaning that the performance increases as the algorithm learns from the data of the subject. This result shows that the adaptation is effective in leading the user toward good performances.

Figure 4 shows the histogram and percent cumulative distribution of the NRD for all 24 subjects and all 12 levels of the Brain Invaders game. The cumulative distribution at the third repetition is 94.44% for the non-adaptive mode and 95.49% for the adaptive mode, that is to say, on the average of all levels and subjects about 95% of the times three or less repetitions suffice to destroy the target. These results

demonstrate that our adaptive system without calibration yields performances equivalent to the traditional system with calibration, already at the first session.

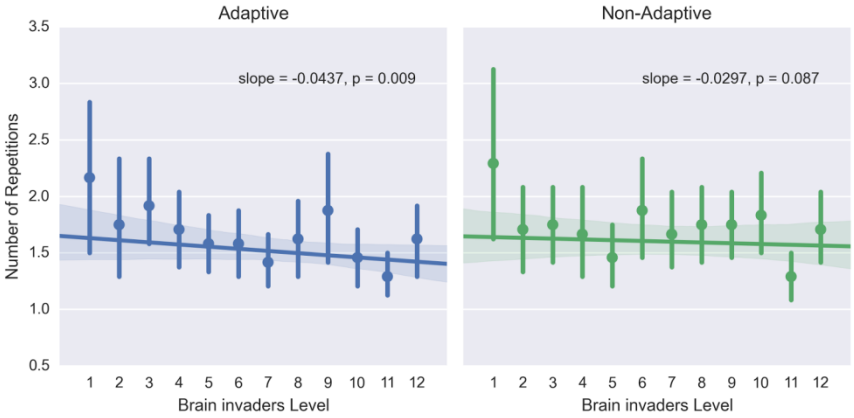

Figure 3: *Adaptation results. Mean (disks) and standard deviation (bars) number of repetitions necessary for destroying the target (NRD) for the 24 subjects across the 12 levels of the first session of Brain Invaders, for the adaptive run (left) and the non-adaptive run (right). On top of the plots is printed the slope of the means and its p-value for the two-tailed test of the slope being significantly different from zero*

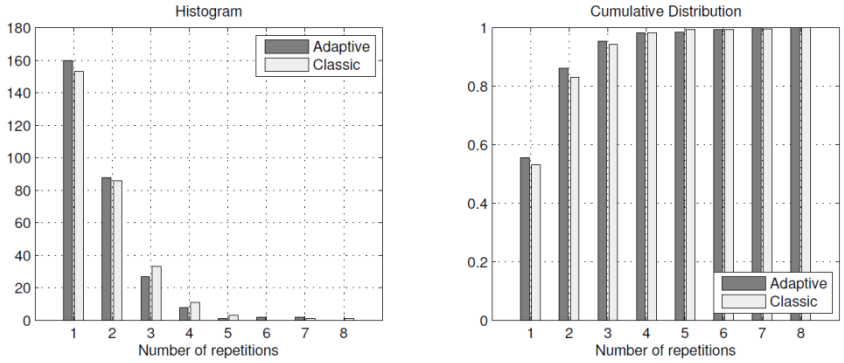

Figure 4: *Comparison of the performance of the adaptive and non-adaptive BCI as a function of the number of repetitions. Raw histogram (left) and percent cumulative distribution (right) of the number of repetitions necessary to destroy the target (NRD) for all 24 subjects and all 12 levels of the first session of the Brain Invaders game*

Implementation

The implementation of the Brain Invaders is achieved with three software modules: acquisition, processing and rendering. Since they communicate to each other via a TCP/IP protocol, they may run on a single computer or on distinct computers in any combination:

– *Acquisition*. This is the OpenViBE acquisition server [REN 2010]. It is in charge of acquiring the data from the EEG machine, streaming the data, correcting for possible amplifiers drifts and sending the data to the OpenViBE platform [REN 2010] (http://openvibe.inria.fr/) for analysis.

– *Processing*. The OpenViBE platform performs data analysis on-line. At the end of each repetition it computes the probability of each alien being the target and sends to the rendering application the indexes of the alien with the highest probability.

– *Rendering*. A dedicated application serves as user interface. The classification results computed by OpenViBE are sent to this application using a VRPN network protocol. Once the result is received in the form of a selected alien, the alien is destroyed on the screen.

Artefact Management

As an option, the Brain Invaders can continuously receive control values from OpenViBE (through a VRPN network protocol). These values can be used for on-line EEG artifact monitoring. Upon reception of a signal flagging the presence of an excessive EEG artifact, we can freeze the Brain Invaders application, display a pause message and wait until a continue signal is received (no EEG artifact is present). We have implemented an on-line artifact monitoring using the Riemannian Potato method [BAR 2013]. In practice, we do not use this feature as pausing the game is annoying for the subject. As a matter of fact the Riemannian MDM method is very efficient, thus its functioning in the presence of small artefacts encountered routinely in real-life experimental sessions is satisfactory.

Brain-Invaders in OpenViBE

Figure 6 shows the workflow implemented in OpenViBE for running the Brain Invaders in adaptive mode. The EEG signal is first filtered in the band-pass region (1-20Hz) using the OpenViBE's *Temporal filter - GPL* box. Then it is down-sampled with a factor of 4 from 512 Hz to 128 Hz thanks to the *Signal Decimation* box. The *Target* and *Non-Target* boxes accept as input the EEG signal and the

triggers. The first uses only triggers that correspond to a flashing group containing the target alien, while the second uses only triggers that correspond to flashing groups non including the target alien. Both boxes output 1-sec epochs of EEG starting at flashing onset (the ERPs) that is provided as input. The *Adaptive MDM* box performs the adaptive classification (the mix of generic and individual classifier as previously explained). The box outputs a decision, that is, the alien to be destroyed, which is sent via the *VRPN Server* box to the Brain Invaders rendering application. The MDM box is implemented in the language Python. Two more OpenViBE boxes exist: *Train MDM* and *Process MDM*. These two boxes are the non-adaptive versions of the *MDM box* (Figure 6). For example they can be used for motor imagery with five movements.

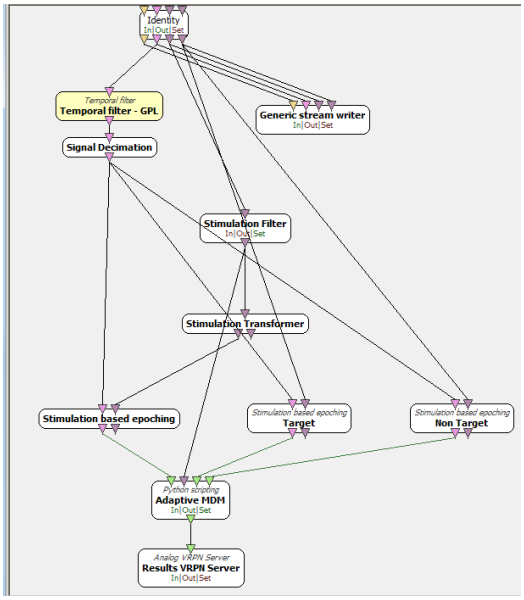

Figure 6: Screenshot of the OpenViBE scenario for running the Brain Invaders in adaptive mode

An additional application called "Brain Invaders Launcher" is also provided. This application configures Brain Invaders and OpenViBE and starts the two automatically for user convenience. The application allows the user to define a certain number of important runtime parameters.

Notes on technical problems

The implementation of P300 BCI-based video games encounters a number of difficulties, the most important one being the *drift* problem. The drift refers to the fact that at least two clocks are involved in the implementation of a BCI-based video-game: the clock of the computer running the user interface (and possibly tagging the data) and the clock of the EEG amplifier. The differences between the paces of the two clocks accumulates over time and results in a larger and larger time difference. In order to tag the EEG data to know the exact stimulation time (when the symbols are flashing) we need to mark the EEG sample corresponding to stimulus onset. This is then needed to extract the (time-locked) event-related potential generated by the stimulus. Sending a flash command to screen and tagging the EEG data stream cannot be executed at the same time. The difference between the two should be as small as possible, but, above all, should be as constant as possible from tag to tag. Also, the time interval between the moment the command is sent and the moment the monitor actually displays the flashes is variable, especially on LCD monitors. The best way to verify the precision of the tagging is to use a light diode sticked on the screen and compare the time difference between the actual flash onset as seen by the diode and the time of the tagging command. The variability of the tag is named the jitter phenomenon. The larger the jitter, the lower the signal-to-noise ratio of averaged ERP and the lower the classification accuracy achievable by the BCI. There are two possible ways to perform data tagging, named here “Hardware Tagging” and “Software Tagging”. In Hardware Tagging the computer running the user interface sends via parallel port a trigger (sort of message) to the EEG machine at the moment of the stimulus presentation. The EEG machine synchronizes the trigger with the flow of incoming EEG data. This type of tagging is very precise (error=+- 2ms in our testing and negligible jitter). Drift is not of concern when Hardware Tagging is used, so the overall jitter is very low in this case. In Software tagging tags are sent internally (by software) from the user interface application to the EEG Data Acquisition Server (DAS) application (For example, the OpenViBE AS). Software Tagging has the advantage of not requiring a cable connection from the rendering application to the EEG machine and of working with any computer and EEG machine, however the overall jitter is much larger (tens of milliseconds on the average at the best according to our tests). When using software tagging the drift problems must be addressed very carefully. In OpenViBE drift can be corrected with the built-in functionality of the OpenViBE AS called “Drift correction”. Unfortunately the current implementation of the drift correction does not work satisfactorily for all EEG amplifiers. Thus, testing of the drift for the available EEG hardware is a mandatory step if software tagging is used.

Use-The-Force!

The second OpenViBE-based video game that we present in this chapter is the game entitled “Use-The-Force”, a game that comes with the OpenViBE platform. The game environment corresponds to the inside of a “Star Wars^{TM} mother ship, in which the player can see a virtual spaceship, namely a Tie-Fighter (see Figure 7). The purpose of the game is to lift the Tie-Fighter up by using the BCI. This task establishes an analogy between the use of the BCI and the use of “the Force” in the Star Wars^{TM} movie. As such, the application was named “Use-The-Force!”. More precisely, the player can lift the Tie-Fighter up by imagining or executing foot movements, those being recognized by the BCI system. The Tie-Fighter is lifted-up at a speed and height proportional to the strength of the Beta ERS (Event Related Synchronization, see chapters 3 and 4) a.k.a., Beta rebound, following the end of the real or imagined foot movement [PFU 1999]. This BCI, its design and properties as well as its OpenViBE implementation are described below.

Figure 7: A user playing with an early prototype of the "Use-the-force" game, in an immersive Virtual Reality room (©Hubert Raguet/Photothèque CNRS) [LOT 2008]

The BCI system

The BCI used for the “Use-the-Force!” is a simple self-paced one. It is based on a single EEG channel (either monopolar or Laplacian), located at position Cz and, as mentioned above, aims at detecting a Beta ERS, appearing posterior to the real or imagined foot movement. To detect this post-movement Beta ERS, a single Band Power (BP) feature is extracted in the Beta band (16-24 Hz) for the last second of

data. This feature is extracted every 100 ms and the last four consecutive features are averaged (with a moving average) in order to produce a smooth Control Signal (CS).

To detect the Beta ERS, and hence, the foot movement, based on the resulting CS, we use a simple threshold Th. If the computed CS is higher than this threshold Th, a foot movement is detected (intentional control state) and a command is sent to the application. If the CS is lower than the threshold Th, the non-control state is detected and no command is sent to the application. This design enables the user to control the BCI in a self-paced way. The value of Th is simply determined according to the mean μ and standard deviation s of a CS epoch obtained while the subject is in a resting state, according to the equation $Th = \mu+3\sigma$. This threshold determination procedure is similar to the one used in another virtual reality application based on BCI [LEE 2007]. It should be noted that Th is determined without using any example of real or imagined foot movement. As such, this BCI does not learn the mental state to be detected.

Evaluation

A first, simple version of the game was evaluated with 21 naïve subjects in a challenging situation: a first-time session, using a single EEG electrode (no Laplacian filter), and during a public exhibition [LOT 2008]. Results showed that, without training, half the subjects could control the game by using real foot movements. A quarter of the subjects could control the spaceship by using imagined foot movements. The results of subjective questionnaires filled out following the system's use showed that the whole application appeared enjoyable and motivating to the users.

A more recent version of the game uses more electrodes, namely electrodes FCz, C1, Cz, C2 and CPz to build a Laplacian derivation over Cz, which leads to improved performances. While this new setup has not been formally evaluated, informal observations suggest that about 90% of naïve users could control the spaceship using real foot movement and more than 50% of them using imagined movements.

Implementation with OpenViBE

In order to design the "Use-the-force!" BCI game with OpenViBE, 2 scenarios are necessary: a first scenario to calibrate the BCI, i.e., to identify the value of the threshold Th to use to detect the post-movement Beta rebound, and a second scenario to detect online this Beta rebound and interact with the 3D game.

The first scenario is represented in Figure 8. It aims at instructing the user to start a resting phase, according to a sound being played and a picture representing a relaxing landscape being displayed, and to measure the mean and standard deviation of the Beta band power of this user in electrode Cz (ideally after Laplacian filtering). The threshold to detect the Beta rebound is then computed as the mean of the Beta

band power at rest plus three times its standard deviation. The instructions to the user are created using the *Lua Stimulator* box, to send OpenViBE stimulations indicating the beginning and end of the rest period, and the *Sound Player* and *Display Cue image* boxes that will play the sound and display the image instructing the user to start/stop resting. The beta Band power in Cz was computed with the *Channel Selector* and *Spatial Filter* boxes to first apply a Laplacian filter, then the *Temporal Filter* box to band-pass filter the data in the 16-24Hz band (i.e., the Beta band), then the *Time-based epoching* (to extract 1s long time windows), *Simple DSP*, *Signal Average* and *Epoch Average* boxes, to square the signal, average it over the 1s time window and average it again over the last 4 time windows, respectively (to smooth the signal).

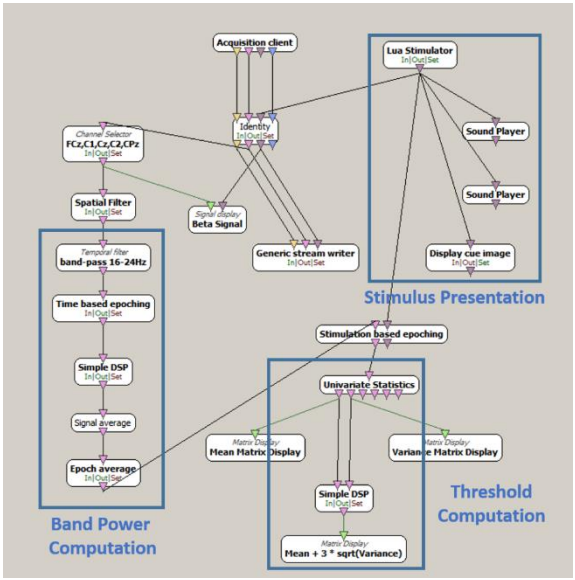

Figure 8: OpenViBE scenario to calibrate the "Use-The-Force!" game, i.e., to compute a threshold on the Beta band power of the user, defined as the mean of this Beta power at rest plus three times its standard deviation. The scenario also includes stimuli (pictures and sounds) to instruct the user when to start and stop resting so that the threshold can be computed.

The Online scenario is displayed in Figure 9. It uses the same boxes as the calibration scenario to compute the smoothed Beta band power over Laplacian channel Cz. It uses new boxes though, first to apply the computed threshold Th to the Beta band power. This is done by cropping the signal below the threshold value (i.e., every band power lower than the threshold value will be set to 0) using the *Crop* box, and shifting it by subtracting the threshold from the resulting signal using

the *Simple DSP* box. This way, the resulting signal – the Control Signal - will be zero when the Beta band power is below the threshold, and positive when it is over the threshold, i.e., when a Beta rebound is detected. This CS is then transmitted to the actual 3D application, using the VRPN protocol. As for "Brain Invaders", the rendering of the game is done in an external application, not part of OpenViBE (although in the case of the "Use-the-Force!" game, this external application is provided with the OpenViBE platform). The OpenViBE designer thus communicates with this external application via VRPN, by sending stimulations (instructions) to this application using the *Button VRPN server* box, which sends button press or button release events according to the received OpenViBE stimulations, and the *Analog VRPN server*, which sends a continuous value to the application, in this case, the CS. On its side, the rendering application moves the spaceship according to the CS it receives via the analog VRPN protocol: if the received CS is zero, the spaceship does not move, if it is positive, the spaceship is lifted from the ground up to an height proportional to the CS. The higher the CS, and thus the bigger the Beta rebound, the higher the spaceship is raised.

Conclusion on Use-The-Force!

The Use-The-Force! BCI-based game is a simple and easy to setup BCI-game based on oscillatory EEG activity. While its gameplay value is limited, it is an interesting and practical demonstration of a BCI-based video game, and therefore a useful starting point to build more advanced and complex BCI games.

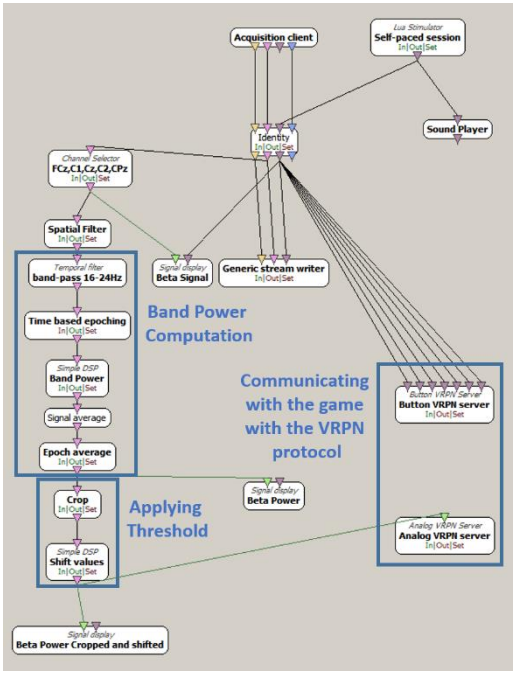

Figure 9: OpenViBE scenario to run the actual use-the-force game, after it is calibrated. This scenario computes the beta band power of the user in the Laplacian Cz channel, threshold it to detect a possible beta rebound following executed or imagined foot movement, and transmit the resulting command to an external 3D application (rendering and animating the spaceship) using the VRPN protocol.

Conclusions

In this chapter we have presented two examples of BCI-based video games: "Brain Invaders" and "Use-The-Force!". We have described their principle, design and characteristics, as well as their implementation with OpenViBE.

"Brain invaders" and "Use-The-Force!" exploit two different kind of brain signals and thus two different kinds of BCI: ERP-based BCI and oscillatory activity (i.e., ERD/ERS) based BCI. As such, we hope they will provide the readers with a useful and potentially inspiring basis to design new and more advanced BCI-games. Regarding "Brain Invaders", our development is open-source and available at: https://bitbucket.org/toncho11/openvibe-gipsa-extensions. The "Use-the-force" game is delivered with the OpenViBE installer and/or code sources, i.e., it is also free and open-source as OpenViBE.

References

[ALL 2003] ALLISON B.Z., PINEDA J.A., "ERPs evoked by different matrix sizes: implications for a brain computer interface (BCI) system", *IEEE Trans Neural Syst Rehabil Eng*, 11(2):110-3, 2003

[BAR 2013] BARACHANT A., ANDREEV A., CONGEDO M., "The Riemannian Potato: an automatic and adaptive artifact detection method for online experiments using Riemannian geometry", *Proceedings of the TOBI Workshop lV*, Sion Switzerland, 2013

[CON 2013] CONGEDO M., EEG Source Analysis, HDR presented at Doctoral School EDISCE, University of Grenoble, 2013

[CON 2013] CONGEDO M., BARACHANT A., ANDREEV A., "A New Generation of Brain-Computer Interface Based on Riemannian Geometry", arXiv:1310.8115, 2013

[CON 2011] CONGEDO M., GOYAT M., TARRIN N., VARNET L., RIVET B., IONESCU G., ET AL, "Brain Invaders: a prototype of an open-source P300-based video game working with the OpenViBE platform", *Proceedings of the 5th Int BCI Conference*, Graz, Austria, 280-283, 2011

[FAR 1988] FARWELL L.A., DONCHIN E., "Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials", *Electroenceph. Clin. Neurophysiol*., 70, 510–23, 1988

[GIB 2008] GIBERT, G., ATTINA, V., MATTOUT, J., MABY, E. & BERTRAND, O., "Size enhancement coupled with intensification of symbols improves P300 Speller accuracy", *4th International Brain-Computer Interface Workshop and Training Course*, Graz, Austria, 2008

[JIN 2011] JIN J., ALLISON B.Z., SELLERS E.W., BRUNNER C., HORKI P., WANG X., NEUPER C., "Optimized stimulus presentation patterns for an event-related potential EEG based brain-computer interface", *Med Biol Eng Comput*, 49(2),181-91, 2011

[LEE 2007] LEEB, R., FRIEDMAN, D., MÜLLER-PUTZ, G. R., SCHERER, R., SLATER, M., PFURTSCHELLER, G. "Self-paced (asynchronous) BCI control of a wheelchair in Virtual Environments: A case study with a tetraplegiac", *Computational Intelligence and Neuroscience*, 2007

[LED 2004] LEDOIT O., WOLF M., "A well-conditioned estimator for large-dimensional covariance matrices", *J Multivar Anal*, 88, 365–411, 2004

[LOT 2007] LOTTE F., CONGEDO M., LÉCUYER A., LAMARCHE F., ARNALDI B., "A review of classification algorithms for EEG-based brain-computer interfaces", *J Neural Eng*, 4(2), pp. R1–R13, 2007

[LOT 2008] LOTTE, F., RENARD, Y., LÉCUYER, A. "Self-paced Brain-Computer Interaction with Virtual Worlds: a Qualitative and Quantitative Study 'Out-of-the-Lab'", *4th International Brain-Computer Interface Workshop and Training Course*, 373-378, 2008

[NIJ 2009] NIJHOLT A., BOS D. P.O., REUDERINK B., "Turning Shortcomings into Challenges: Brain-Computer Interfaces for Games, *Entertainment Computing*, 85-94, 2009

[PFU 1999] PFURTSCHELLER, G., DA SILVA, F. H. L. "Event-related EEG/MEG synchronization and desynchronization: basic principles", *Clinical Neurophysiology*, vol 110, 1842-1857, 1999

[REN 2010] RENARD Y., LOTTE F., GIBERT G., CONGEDO M., MABY E., DELANNOY V., BERTRAND O., LÉCUYER A., "OpenViBE: An Open-Source Software Platform to Design, Test and Use Brain-Computer Interfaces in Real and Virtual Environments", *Presence: Teleoperators and Virtual Environments*, 19(1), 35-53, 2010

[RIV 2009] RIVET B., SOULOUMIAC A., ATTINA V., GIBERT G., "xDAWN algorithm to enhance evoked potentials: application to brain-computer interface", *IEEE Trans Biomed Eng*, 56(8):2035-43, 2009

[TOW 2010] TOWNSEND G., LAPALLO B.K., BOULAY C.B., KRUSIENSKI D.J., FRYE G.E., HAUSER C.K., ET AL., "A novel P300-based brain-computer interface stimulus presentation paradigm: moving beyond rows and columns", *Clinical Neurophysiology*, 121(7), 1109-20, 2010

[VID 2009] VIDAURRE C., KRÄMER N., BLANKERTZ B., SCHLÖGL A., "Time domain parameters as a feature for eeg-based brain computer interfaces", *Neural Networks*, vol. 22, 1313–1319, 2009

[WOL 2011] WOLPAW J., WOLPAW E.W., *Brain-Computer Interfaces: Principles and Practice*, Oxford University Press, Oxford, 2011