STIPRESOFT: an alternative stimuli presentation software synchronizing with current acquisition systems in EEG experiments

Abdurrahman Özbeyaz¹

Received: 18 August 2019 / Accepted: 13 November 2019 / Published online: 19 November 2019
© Springer Nature Switzerland AG 2019

Abstract

Since analyzing of visual evoked potentials is crucial in neuropsychology experiments, stimuli presentation software should be meticulously designed. In such experiments, synchronizing stimuli presentation software with existing biomedical instruments, and marking EEG signals are difficult problems to overcome. In EEG experiments, these kinds of problems have been still tried to be overcome with the simple and unprofessional software like PowerPoint. Furthermore, commercial software is costly and may be incompatible with the current laboratory resources. In this study, free and simple STIPRESOFT software which could synchronize stimuli presentation software with available EEG devices and could mark EEG signals at the time of visual stimulus was developed. In the applied experiments, the time latency between the stimulus and marking times were more successful with 0.028 ms according to the competitors. In the software, the advantages of the Win-API were benefited. In STIPRESOFT, two API were exploited to improve timing precision, and to enable synchronization. Furthermore, time stamps related to stimuli on EEG signals were recorded in a text file during the experiment. Consequently, the developed software presents an alternative free and simple solution for researchers who need to adjust synchronization between stimuli presentation and existing EEG device in the scientific studies.

Keywords EEG · Visual stimuli · Presentation software · Synchronization · Win-API

1 Introduction

An electroencephalogram (EEG) is the brain activity acquired from the electrodes placed on the brain surface. By analyzing these signals, meaningful results can be extracted from the electric activities performed in the brain. EEG signals have been widely used in the diagnosis and treatment of some special brain and neurological diseases [1, 2]. Some EEG experiments have been performed against visual, auditory or somatosensory stimuli in the literature. Thus, communications between people and their surroundings have been explored, and some brain activations have been observed. The evoked potential (EP) obtained from the brain is a superposition of various electrical activities. EP is the wave found in the EEG and arises

in the brain as a result of neural activities in response to a stimulus. Visual evoked potentials (VEPs) are employed in experimental studies for the determination of the visual functions [3]. When a visual expression is used as a stimulus, some descriptive information can be reached by classifying the EP [4, 5].

Some medical EEG devices may not be sufficient for organizing stimuli experiments. Moreover, although synchronizing EEG devices with stimuli presentation software is crucial, existing devices may not have the capability to provide this feature [6]. Furthermore, timing accuracy is important in the stimuli presentation [7]. In some scientific EEG studies, researchers are still using unprofessional software like PowerPoint. Such types of presentation software cannot synchronize with existing EEG devices and

✉ Abdurrahman Özbeyaz, aozbeyaz@adiyaman.edu.tr | ¹Department of Electrical Electronics Engineering, Engineering Faculty, University of Adiyaman, Adiyaman, Turkey.

cannot mark on signals in EEG experiments. There are lots of commercial software for EEG studies, and researchers are obliged to purchase such commercial software due to the inadequacies of unprofessional software. However, these types of software are not preferred because of high prices and incompatibility with existing EEG devices.

In some previous studies, non-specialized presentation software was necessarily used due to scarce project budget resources while performing EEG experiments. In some of these studies, automated and PC-based systems were designed for performing EEG experiments [3, 8]. And in an experiment, visual evoked potentials were recorded using a special hardware system [9]. In another study, researchers investigated fixation differences between familiar and unfamiliar face stimuli using non-specialized presentation software [10]. In another EEG experiment [11], similar-sized, high-quality shoe photographs were presented using PowerPoint. Furthermore, familiar and unfamiliar faces were displayed to subjects using non-specialized presentation software in an EEG study [12]. In another study in which animal and non-animal natural scenes were shown, PowerPoint was employed as stimuli presentation software [13]. Apart from these, non-free and commercial stimuli presentation software like E-Prime [14–16] and STIM2 [17, 18] can be used in EEG experiments. Moreover, free but complicated software like PsychoPy [19] and OpenSeSama [20] can be preferred for building and running stimuli experiments in neurophysiology studies [21]. Because of their cost and complexity, these types of software may not be preferred.

As an alternative to the aforementioned software, a novel, simple and free stimuli presentation software STIPRESOFT was developed in this study. The motivation behind the study was to set up suitable, practical EEG experiments in which visual stimuli were used without the need for any additional budget. Furthermore, the developed software momentarily synchronized the EEG acquisition system with the stimuli presentation software. Besides, the mechanism of the software had good timing accuracy. In the software, small time shifts between displaying and recorded times of a stimulus were eliminated thanks to the API functions. The software was entirely free, easy to use, and had support for existing hardware. The software could make direct calls to the Windows Application Processing Interface (API) libraries. This software was simple to run on an extremely wide range of hardware like Nicolet v32/v44, Nihon Kohden EEG 1100 and NeuroFax EEG Systems. The developed software aimed to contribute to similar studies for facilitating EEG experiments and to reduce software costs.

2 Materials and methods

In this study, STIPRESOFT software was developed. This software could synchronize the existing EEG device with a stimuli presentation software. The developed software was experienced on the EEG device registered on the inventory of the Adiyaman University Training and Research Hospital Neurology service.

2.1 Hardware specifications

The EEG device was a seamless, reliable and flexible system that recorded brain wave activities and electroencephalogram signals without compromising quality. The EEG system combined photic stimulation, counting for hyperventilation, re-organization of data, reporting and many other features in one system. Furthermore, this system supported optional add-on packages such as digital video, sharp wave detection, seizure detection and sleep analysis. The device had an integrated impedance interface, a SpO2 port, an Ethernet amplifier interface, nine AC and DC ports, one high-level DC input port and a patient response button. This response button was not used in the study.

2.2 EEG recording software

The recording software (Studyroom) managed the job of obtaining the EEG signals. Various controls related to EEG experiments could be adjusted by this software. Furthermore, the recording process could be started by pressing the RECORD button or CTRL + R key combination in the software. This software was synchronized with STIPRESOFT during the experiment. STIPRESOFT provided the synchronization by sending CTRL + R key combination to the Studyroom software at the onset and end of the experiment.

2.3 Experimental conditions

The software randomly reflected the stimuli to the monitor in each run. Display sequences and display times for stimuli and experiment duration were recorded in a text file (behavioral). The experimental procedure was organized according to the method tried in [22]. A total of 100 cycles were processed on each run. The recording time was 9 min and 10 s for each run. The developed software in the study is convenient to be run under different experimental conditions.

2.4 Windows application processing interface (windows API)

Operating systems service their sub-processes, and this software is interrupted with Windows APIs. To develop software with APIs, the basic features and the functions served by the operating system must be known to the developer. Interactions among Windows forms, like sending a button click event from a form to another form, are possible using Windows API functions. API functions benefit from DLL files. To use these functions in another project, the corresponding DLL files have to be initially included. In this study, the form control properties of Windows API functions were employed.

2.5 Timer object

The timer object is a precise function. However, the processes defined by a timer object may cause time shifts in the more exacting time-dependent operations. Namely, when a timer object is employed in an image-processing application, values such as physical sizes, components of images and writing to a screen may cause time delays. In case there is a need for advanced precision time measurements, *Query-Performance-Frequency* and *Query-Performance-Counter* components can be benefited. The sensitivities of time-oriented functions used in the programming languages are shown in Table 1.

2.6 Query-performance-counter (QPC)

The total number of ticks is returned by Query-Performance-Counter (QPC) in the Windows operating system. QPC is independent of any external time references. Time stamps and time interval calculations are an integral part of a computer system. In computer systems, the performance calculations include the computation of response time, throughput and latency. These operations involve the calculation of processes executed throughout a time interval that is defined by a start event and an end event. Consequently, QPC calculates the small-time intervals in a computer system.

Table 1 Time-oriented functions used in programming

Functions	Unit	Sensitivity
Now, time, timer	Second	1 s
GetTickCount	Millisecond	Approximately 10 ms
TimeGetTime	Millisecond	Approximately 10 ms
QueryPerformanceCounter	QueryPerformanceFrequency	Real time

Fig. 1 A screenshot belongs to GUI providing synchronization

2.7 Query-performance-frequency (QPF)

Query-Performance-Frequency (QPF) is a function dependent on QPC. This function gives the frequency of the performance counter in a computer system. The frequency of the performance counter is set when the system boots and is stable at all processor times. Hence, the frequency amount is queried at the instant of the initialization of applications, and the results obtained are cached by the computer. In a computer system, the period and resolution are equal to the reciprocal of the frequency value. The performance counter frequency is determined at system initialization and does not change while the system is running.

3 Results

In the study, the software enabling synchronization between the stimuli presentation software and the EEG recording system was developed using Windows API functions. The developed software can be accessed from the address given in [23]. The visual results obtained, and software codes developed in the study are given in the following.

3.1 Graphical user interfaces (GUI)

Two graphical user interfaces (GUI) were developed through the study. One of them was the synchronization test program. In this GUI, a button aims to list the names on the Windows forms belonging to the applications running in the background. Another button sends a CTRL + R key combination to the EEG recording software. A screenshot belonging to this GUI is given in Fig. 1.

Another GUI was the main stimuli presentation software. Synchronization was provided by this GUI.

Fig. 2 Developed GUI (a), experimental procedure (b)

Furthermore, this software sent the CTRL + R key combination through API to Studyroom software (EEG acquisition system) to start and stop the recording of EEG signals. In the first, send of the key combination started the EEG recording, and in the second, send stopped the EEG recording. Time stamps for the events occurring in the presentation were saved in a text file in each run. The decomposition of EEG signals according to the events was possible through the help of the text file. Moreover, event-related potential (ERP) could be obtained according to the instantaneous stimuli displays recorded in this file. A screenshot related to this GUI and the graphical abstract belonging to the experimental procedure are given in Fig. 2a, b, respectively.

3.2 Source codes

In the STIPRESOFT software, there are three sub-functions and one main function. The names of the sub-functions are WindowFormLists, FormSynchronization and PressCTRLwithKey, respectively. These sub-functions are introduced under the headings of code snippet 1, code snippet 2 and code snippet 3 in this section. And finally, the main function is introduced under the heading of code snippet 4. The source codes of these functions are given in “Appendix” section, and their flowcharts are given in Figs. 3, 4, 5 and 6. The codes can be followed up from the line numbers in “Appendix” section.

The first procedure and the related flowchart of the developed software are given in code snippet 1 and Fig. 3. In the code snippet 1, the *WindowFormLists* procedure lists the form names of applications running at that moment in the background. The *IsWindowVisible* command examines the forms if they are running in the background of the operating system. The *GetWindowText* function copies the text of the specified window's title bar into a buffer. The *GetClassName* retrieves the name of the class to which the specified window belongs, and the *GetnextWindow*

Fig. 3 The function listing the names of Windows forms in the application

Fig. 4 The function that sends shortcut key combination to another window forms for assuring EEG signals recording

retrieves a handle to the next or previous window in the Z-Order.

In code snippet 2, the *FormSynchronization* function sends the CTRL + R key combination to the EEG recording interface. This interface is handled by the *WindowFormLists* procedure given in code snippet 1. The

Fig. 5 The function for sending key combination to synchronized form

Fig. 6 The main function of the presentation software

API Overhead = 5 Freq = 3579545 iFirst = 22256710076 Start = 6217748,3649458243 Finish = 09:24:47 (+) "6217748"

System Time	Starting Time	Sample	Calculated Time	Stimuli
09:25:52	09:24:47	65005	09:25:52	GrayFix
09:25:54	09:24:47	67507	09:25:54	Picture
09:25:55	09:24:47	68301	09:25:55	Gray
09:25:56	09:24:47	69505	09:25:56	GrayFix
09:25:59	09:24:47	72003	09:25:59	Picture
09:25:59	09:24:47	72802	09:25:59	Gray
09:26:01	09:24:47	74005	09:26:01	GrayFix
09:26:03	09:24:47	76505	09:26:03	Picture
09:26:04	09:24:47	77310	09:26:04	Gray
09:26:05	09:24:47	78510	09:26:05	GrayFix

Fig. 7 A screenshot of a sample text file. Time stamps of displayed stimuli images are saved in the file

Fig. 8 Mean square errors (MSE) for time latencies of 120 stimuli in different sessions

The *PressCTRLwithKey* function behaves as the CTRL + ANY KEY shortcut combination. The source codes related to this function are given in code snippet 3. The flowchart belonging to code snippet 3 is given in Fig. 5.

The main function in STIPRESOFT is given in code snippet 4. In the first paragraph (the first 19 lines), in code snippet 4, a text file is initially generated to record the presentation times of the stimuli images. This text file makes it possible for the decomposed EEG signals to be related to the stimuli images. Between 10 and 12 lines, QPF and QPC functions record the times related to these events. The *FormSynchronization* function given in line 17 sends the CTRL + R key combination to the window form belonging to the EEG acquisition system to start synchronization between the two forms. In the last paragraph (lines between 20 and 59), in code snippet 4, cycles are repeated, and event times are recorded into the text file for each while loop. The flowchart designed for the main function is given in Fig. 6.

In the software development process, the study was confronted by a problem related to the use of the timer object. The timer object did not act consistently in the study. Namely, the *Sleep* function inherited from the timer object was employed to maintain images on a screen for the desired time. However, this function could not maintain the images on the screen for the wanted time during the presentation. Time shifts were also confronted in this application. For instance, it was observed that the screen retention times for stimuli images might be 798 or 814 ms instead of 800 ms in some trials. Since this negativity affected the synchronization between windows forms, Windows API functions (QPF, QPC) were employed instead

of the *Sleep* function. These functions achieved precise timing adjustments for the presentation.

Time stamps recorded during an experiment are given in Fig. 7. Response times (GrayFix, Gray and Picture) are shown for only three stimuli presentation time in the figure. The high-resolution timing functions (QPF and QPC) are employed from the beginning of the while loop. The loop continues through the end of the experiment. During the loop, cycles are displayed one after another. One loop includes a fixation providing focusing. Throughout the loop, all events in time are recorded for each repetition.

The *iFirst* and *iSecond* variables keep the total number of ticks that have occurred since the Windows operating system was started. The API Overhead, which is the delay between two consecutive queries, is calculated with the *iDifference* variable, and the *iFrequency* variable is defined to get the frequency value for microprocessors. The *dResult* is a variable that gives the time value in milliseconds. The *Application.ProcessMessages* command allows a program to become operational and perform other applications when a long loop like this is active in the software. The loop ends after all visual stimulus images have been displayed to the subjects. At the end of the given codes, the *FormSynchronization* function is recalled to terminate the synchronization between two software programs.

In the study, the quantitative time latencies were measured over 120 stimuli. According to the data obtained in three sessions, time latencies were measured as 0.0098, 0.0106 and 0.0085 ms averagely. And standard deviations of time latencies were 0.0082, 0.0106 and 0.0073, respectively, in each session. The results related to measured time latency are given in Table 2 and Fig. 8.

Table 2 Statistics about time latencies for 120 stimuli presentation

	Session 1	Session 2	Session 3
Mean	0.0098	0.0106	0.0085
STD	0.0082	0.0106	0.0073
Variance	6.69E-05	0.00011	5.33E-05
MSE	0.00016	0.00022	0.00013
RMSE	0.01277	0.01496	0.01118

4 Discussions

In a lot of EEG researches, synchronization between stimuli presentation and EEG acquisition systems has always been a major concern when analyzing EEG signals, and for this reason, the extraction process for the EP is difficult. Therefore, experiments have to be well designed in EEG studies. In the published literature, while some EEG experiments are designed using non-specialized applications [10–13], some [14–17, 20, 24] are designed to use commercial and non-free software to meet stimuli presentation requirement. In this study, alternative software was developed to present stimuli. Moreover, this software enabled synchronization between stimuli presentation software and EEG recording software in the EEG experiments. While it was observed that the average time between the stimulus presentation and marking process is 2.27 ms, and the standard deviation was 0.70 ms in the commercial software, these values were 0.028 and 0.026 ms in the developed software, respectively. It was seen that the average time latency between stimuli presentation and the time for a marker was more precise in the proposed software. From this point of view, the study brought innovation to the field. Moreover, comparisons of the proposed software with the other competitors are given in Table 3. The features in the table are prepared using the web page of other competitors.

In the experimental design, two main problems were confronted. One of them was the time accuracy problem occurring during the screening of stimuli, and the other was the time synchronization problem between stimuli

presentation and EEG acquisition system. In previous studies, these problems were overcome either by using commercial software and devices or by some non-specialized methods. While commercial software was a burden to project budgets, non-specialized methods cause a lot of additional tiring works. In our study, these problems were overcome with the Windows API functions. The time accuracy problem was that the stimuli images could not be shown for the desired times on a screen. Because of this problem, time shifts occurred in the signals. The time accuracy problem was overcome using two Windows API functions (QPF, QPC). These functions enabled the time duration to be set in the frequency band. Thereby, more precise timing procedures were performed in the experiments. The second problem was the synchronization of the reaction times of stimuli presentation and EEG recording software. This problem was overcome with a newly developed function (*PressCTRLwithKey*) as a result of this study. Such that CTRL + R key combination was sent from STIPRESOFT to the EEG signal acquisition software at the start and the stop time of the experiment, and display times for stimuli images were saved to a text file for each run in the experiment.

Hardware and stimuli dependencies are crucial for the software used in neuroscience and psychology studies. Therefore, the developed software in the study has been designed as flexible in terms of hardware and stimuli types. Investigators can use other stimuli types instead of visual stimuli with this software. Moreover, STIPRESOFT can be integrated with EEG devices, which can be used for other projects. Accessing an input/output device from the developed software can be allowed. In our experimental procedure, the software has been used for sending the CTRL + R key combination between two computer systems. The software is also flexible enough to send other key combinations to the acquisition system. In neuroscience studies, timing is a critical issue for many experiments. Temporal precision is required to within a few milliseconds, or even in the sub-millisecond range [19]. The developed software presents a robust method to achieve very precise timing of events and to synchronize with other devices.

Table 3 Comparisons of STIPRESOFT software with the other competitors

	STIPRESOFT	E-Prime	STIM2	OpenSesame	PsychoPy
Free	✓	✗	✗	✓	✓
Full source code	✓	✗	✗	✓	✓
Synchronization with existing hardware	✓✓	✗	✗	✓	✗
Stimuli in real time	✓	✓	✓	✓	✓
Timing precision	✓✓	✓	✗	✓	✗
Ease and use of installation	✓	✓	✓	✓	✗

5 Conclusions

The developed synchronization-based stimuli presentation software (STIPRESOFT) in the study can be accessed from [23] for EEG studies, including stimuli presentations. This software can be integrated into the existing EEG devices in hand freely. Furthermore, when the developed software is employed, there is no need to use extra equipment in the EEG experiment. This software aims either to reflect the visual stimuli on the screen in random order or to synchronize the presentation times of stimuli with the times of signal recordings during the experiment. Through the software, signals can be easily decomposed according to the time stamps recorded in a behavioral file. Moreover, the developed software is novel and simple in terms of complexity compared to software that are provided free of charge [21] in the literature, and it is unpaid compared to commercial ones. And the proposed software is outperformed in terms of time latencies according to the competitors.

Consequently, the proposed software in this study is a comprehensive stimuli presentation software that can employ the Windows API library's functions for neuropsychological experimental tasks. It is a modern tool providing well-defined paradigms that may be widely used, integrating with the existing acquisition system fully and providing synchronized trigger pulses.

Acknowledgements I would like to thank the staff of the Neurology Department of Adiyaman University, Adiyaman, Turkey, who has provided the necessary equipment and environment.

Compliance with ethical standards

Conflict of interest The author declares that he has no conflict of interest.

Ethical standards All procedures performed in the studies involving human participants were by the ethical standards of the institutional research committee and with the 1964 Declaration of Helsinki and its later amendments or comparable ethical standards.

Informed consent Informed consent was obtained from all individual participants included in the study.

Appendix

```

procedure WindowFormLists (Hand : Hwnd; Liste:TStrings);
var
    Name,Title,ClassName : array[0..255] of char;
begin
    while Hand > 0 do
        begin
            if IsWindowVisible (Hand) then
                begin
                    GetWindowText (Hand,Title ,255);
                    GetClassName (Hand, ClassName,255);
                    Listbox.Add (Format ('%8d : %s ( %s )', [Hand, String (Title),
String (ClassName)] ) );
                end;
                Hand := GetnextWindow (Hand,GW_HWNDNEXT);
            end; // While
        end;
end;

```

Code Snippet 1. The procedure listing the names of Windows forms in the application.


```
procedure FormSynchronization;  
var  
    hHandeled: THandle;  
    WindowName: PAnsiChar;  
begin  
    WindowName:= PAnsiChar(Edit3.Text);  
    hHandeled := FindWindow('NicOneRecorder', WindowName);  
    If hHandeled <> 0 then  
        begin  
            SetForegroundWindow(hHandeled);  
            PressCTRLwithKey(Ord('R'));  
        end  
    end;  
end;
```

Code Snippet 2. The procedure that sends shortcut key combination to another window forms for assuring EEG signals recording.

```
procedure PressCTRLwithKey(iKey : Integer);  
begin  
    keybd_event(VK_CONTROL, MapVirtualKey(VK_CONTROL, 0), 0, 0);  
    keybd_event(iKey, MapVirtualKey(iKey, 0), 0, 0);  
    keybd_event(iKey, MapVirtualKey(iKey, 0), KEYEVENTF_KEYUP, 0);  
    keybd_event(VK_CONTROL, MapVirtualKey(VK_CONTROL, 0), KEYEVENTF_KEYUP, 0)  
end;
```

Code Snippet 3. The procedure for sending key combination to synchronized form.

```
1 procedure Main_Function();
2 begin
3   k := 65000; j := 0;
4
5   AssignFile(myfile, 'E:\'+txtNameSurname.Text+'.txt');
6   Append(myfile);
7   Writeln(myfile, TimeToStr(t2));
8   CloseFile(myfile);
9
10  QueryPerformanceFrequency(iFrequency);
11  QueryPerformanceCounter(iStart1);
12  QueryPerformanceCounter(iStart2);
13  iDifference := iStart2-iStart1;
14  dStart := ((iStart1 - iDifference) / iFrequency) * 1000;
15  T2 := Now();
16
17  FormSynchronization();
18
19  Image1.Picture.LoadFromFile('./fix/blackwhitecheckboard.jpg');
20  while j <= 119 do
21  begin
22    QueryPerformanceCounter(iFirst);
23    QueryPerformanceCounter(iSecond);
24    iDifference := iSecond - iFirst;
25    dResult := ((iFirst - iStart1- iDifference) / iFrequency) * 1000 ;
26    T3 := IncMilliSecond(t2, Trunc(dResult));
27
28    if ((Trunc(dResult) >= 5000) and (xStart = False)) then
29    begin
30      Image1.Picture.LoadFromFile('./fix/start.bmp');
31      xStart := True;
32    end
33    else if Trunc(dResult) >= k then
34    begin
35      if ((Trunc(dResult) >= k) and (xGrayFix = False)) then
36      begin
37        Image1.Picture.LoadFromFile('./fix/grayfix.bmp');
38        xGrayFix := True;
39        xPicture := False;
40      end
41      else if ((Trunc(dResult) >= k+2500) and (xPicture = False)) then
42      begin
43        Image1.Picture.LoadFromFile('./images/'+stimuli[j]+' .jpg');
44        xPicture := True;
45        xGray := False;
46      end
47      else if ((Trunc(dResult) >= k+3300) and (xGray = False)) then
48      begin
49        Image1.Picture.LoadFromFile('./fix/gray.bmp');
50        xGray := True;
51        xGrayFix := False;
52        k := k + 4500;
53        j := j+1;
54      end;
55    end;
56    Application.ProcessMessages;
57  end; // While end
58  FormSynchronization;
59 end;
```

Code Snippet 4. The main procedure of the presentation software.

References

- Jasper HH (1937) Electrical signs of cortical activity. *Psychol Bull* 34:411–481. <https://doi.org/10.1037/h0053401>
- Adrian ED, Yamagiwa K (1935) The origin of the berger rhythm. *Brain* 58:323–351. <https://doi.org/10.1093/brain/58.3.323>
- Micheli-Tzanakou E (1989) A PC-based system for visual evoked potential studies. *J Med Syst* 13:31–35
- Sutter EE (1992) The brain response interface: communication through visually-induced electrical brain responses. *J Microcomput Appl* 15:31–45. [https://doi.org/10.1016/0745-7138\(92\)90045-7](https://doi.org/10.1016/0745-7138(92)90045-7)
- Murphy JT (1996) Spehlmann's evoked potential primer visual, auditory and somatosensory evoked potentials in clinical diagnosis. *Can J Neurol Sci* 23:84. <https://doi.org/10.1017/S031716710003924X>
- De Clercq A, Crombez G, Buysse A, Roeyers H (2003) A simple and sensitive method to measure timing accuracy. *Behav Res Methods Instrum Comput* 35:109–115
- Plong M, Shen K, Vliet M Van, Robben A, Hulle M Van, Geurts L (2012) Accurate visual stimulus presentation software for EEG experiments. In: Asian conference on information systems, pp 1–4
- Micheli-Tzanakou E (1984) An automated system for visual studies. *J Med Syst* 8:507–519
- Moreno L, Sánchez JL, Mañas S, Piñeiro JD, Merino JJ, Sigut J, Aguilar RM, Estévez JI, Marichal R (2001) Tools for acquisition, processing and knowledge-based diagnostic of the electroencephalogram and visual evoked potentials. *J Med Syst* 25:177–194. <https://doi.org/10.1023/A:1010780900068>
- van Belle G (2010) Fixation patterns during recognition of personally familiar and unfamiliar faces. *Front Psychol* 1:1–8. <https://doi.org/10.3389/fpsyg.2010.00020>
- Yılmaz B, Korkmaz S, Arslan DB, Güngör E, Asyalı MH (2014) Like/dislike analysis using EEG: Determination of most discriminative channels and frequencies. *Comput Methods Prog Biomed* 113:705–713. <https://doi.org/10.1016/j.cmpb.2013.11.010>
- Çelik U, Arica S (2013) Classification of evoked potentials of familiar and unfamiliar face stimuli using multi-resolution approximation based on excitatory post-synaptic potential waveform. *Comput Electr Eng* 39:1571–1584. <https://doi.org/10.1016/j.compeleceng.2013.03.009>
- Fabre-Thorpe M, Delorme A, Marlot C, Thorpe S (2001) A limit to the speed of processing in ultra-rapid visual categorization of novel natural scenes. *J Cogn Neurosci* 13:171–180
- Barbeau EJ, Taylor MJ, Regis J, Marquis P, Chauvel P, Liegeois-Chauvel C (2008) Spatio temporal dynamics of face recognition. *Cereb Cortex* 18:997–1009. <https://doi.org/10.1093/cercor/bhm140>
- Genetti M, Khateb A, Heinzer S, Michel CM, Pegna AJ (2009) Temporal dynamics of awareness for facial identity revealed with ERP. *Brain Cogn* 69:296–305. <https://doi.org/10.1016/j.bandc.2008.08.008>
- Wirsich J, Bénar C, Ranjeva J-P, Descoins M, Soulier E, Le Troter A, Confort-Gouny S, Liégeois-Chauvel C, Guye M (2014) Single-trial EEG-informed fMRI reveals spatial dependency of BOLD signal on early and late IC-ERP amplitudes during face recognition. *Neuroimage* 100:325–336. <https://doi.org/10.1016/j.neuroimage.2014.05.075>
- Gopalakrishnan R, Machado AG, Burgess RC, Mosher JC (2013) The use of contact heat evoked potential stimulator (CHEPS) in magnetoencephalography for pain research. *J Neurosci Methods* 220:55–63. <https://doi.org/10.1016/j.jneumeth.2013.08.015>
- Gao J, Lin P, Yang Y, Wang P, Zheng C (2010) Real-time removal of ocular artifacts from EEG based on independent component analysis and manifold learning. *Neural Comput Appl* 19:1217–1226. <https://doi.org/10.1007/s00521-010-0370-z>
- Peirce JW (2008) Generating stimuli for neuroscience using PsychoPy. *Front Neuroinform* 2:1–8. <https://doi.org/10.3389/neuro.11.010.2008>
- Lohse M, Overgaard M (2017) Emotional priming depends on the degree of conscious experience. *Neuropsychologia*. <https://doi.org/10.1016/j.neuropsychologia.2017.10.028>
- Ruisoto P, Bellido A, Ruiz J, Juanes JA (2016) Building computer-based experiments in psychology without programming skills. *J Med Syst* 40:148. <https://doi.org/10.1007/s10916-016-0511-8>
- Eimer M (2000) Effects of face inversion on the structural encoding and recognition of faces. Evidence from event-related brain potentials. *Brain Res Cogn Brain Res* 10:145–158
- Ozbeyaz A (2018) Stimuli presentation software. In: Adiyaman University. <http://ozbeyaz.adiyaman.edu.tr/software.php>. Accessed 7 Marh 2018
- Peirce JW (2007) PsychoPy—psychophysics software in python. *J Neurosci Methods* 162:8–13. <https://doi.org/10.1016/j.jneumeth.2006.11.017>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.