Center Speller (008-2015)

Matthias Treder, Benjamin Blankertz
Technische Universität Berlin, Berlin, Germany

September 5, 2016

1 Introduction

The aim of this study was to develop a visual speller that does not require eye movements. Like the Hex-o-Spell, the Center Speller operates in two levels, selecting a letter group in the first level and the desired letter from the selected letter group in the second level. The Center Speller uses unique geometric shapes, each one having a unique colour. Shapes are presented centrally in a sequential fashion. We conceive of the selection process in the Center Speller as non-spatial. It is true that participants attend to the spatial location at which stimuli are presented. However, spatial attention alone enhances the response to all stimuli presented at the attended location, that is, both targets and nontargets. It is therefore the visual features (colours and shapes) that entail attentional selection.

We performed an online experiment with 13 healthy participants. All participants achieved highly accurate BCI control. They could select one out of thirty symbols (chance level 3.3%) with mean accuracies of 97.1% for the Center speller. For detailed information on the experiment and the method, refer to [1].

2 Data

For details on data pre-processing, refer to the papers in the reference list. The data has been pre-processed with the BBCI Matlab toolbox (`https://github.com/bbci/bbci_public/`). For an introduction to the toolbox, refer to `https://github.com/bbci/bbci_public/blob/master/doc/index.markdown`. Note that the data can easily be transformed into FieldTrip, EEGLab, and similar data types, although this needs to be done manually as currently there are no automated scripts for this.

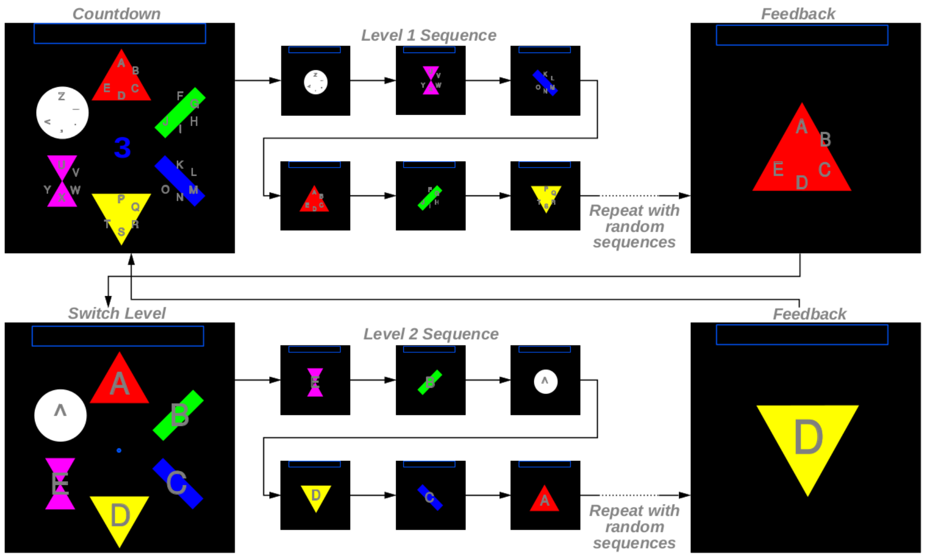

Figure 1: Center Speller. A selection starts with a countdown during which all the letter groups and the corresponding symbols are shown at different spatial locations. Note that during the sequence all stimuli are shown sequentially at the same central location. In level 1, the subject selects the target letter group by attending to the corresponding symbol. When the letter group has been selected, the letters are assigned to the different symbols. Again, the subject focuses on a particular symbol to finally select the desired letter.

3 General description of Matlab structs

`data` contains the EEG data. `mrk` contains timing information about the events (i.e. markers/triggers). `mnt` contains details about the montage useful for plotting including channel positions in 2D and 3D.

- `data`
  - `X`: matrix of size [number of time points × number of channels]. Contains the continuous EEG data.
  - `fs`: Sampling frequency in Hz.
  - `clab`: Cell array with channel labels.
  - `trial`: time of events in samples relative to the start of the measurement. Can be transformed into milliseconds by `data.trial/data.fs * 1000`.
  - `classes`: Cell array with labels for the different types of events.
  - `y`: Vector of the same length as `trial` indicating, for each trial, which class it belongs to. For instance, if the first 3 entries of `y` are 1, 3, 2, then the first trial belongs to class 1, the second trial belongs to class 3, and the third trial belongs to class 2. In rare occasions, if an entry is 0, it means there is no trial information for this particular event and it should probably be discarded.
- `mrk`
  - `time`: time of events in milliseconds relative to the start of the measurement.
  - `y`: logical matrix of size [number of classes × number of events]. It contains the same information as `data.y`, but the format is different. Each row of `mrk.y` corresponds to one experimental condition where 1's indicate that the corresponding trial belongs to this condition. Indices correspond to the indices in the `data` struct, that is for the k-th class, `find(mrk.y(k,:))` and `find(data.y == k)` give equal results.
  - `className`: same as `data.classes`
  - `event`: The event field contains additional information for each of the events

  The `mrk` structure usually contains further sub-structures which have the same structure as the `mrk` struct. These substructures usually carry additional information taken from additional event triggers that were recorded.
- `mnt`
  - `x`: x-position of electrode for 2D plotting
  - `y`: y-position of electrode for 2D plotting
  - `pos_3d`: 3D-position of electrode for 3D plotting

4 Selecting events from `data`

If the data contains different classes/experimental conditions, the event indices corresponding to the first class can be selected using `find(data.y == 1)`. Then, `find(data.y == 2)` yields the event indices for the second class, and so on. The indices can be saved in a variable and then used to obtain the onset times of trials corresponding to the selected class. The onsets can then be used for epoching the data. Example code: `idx1 = find(data.y == 1); trialOnsetsClass1 = data.trial(idx1);`

5 Calibration data and online data

For each subject we provide two datasets. `data{1}` contains the calibration phase where training data was collected. `data{2}` contains the online data in the copy-spelling and the free-spelling mode. To simulate online operation, a classifier can be trained using the calibration data and be evaluated using the online data.

Note that in `data.classes` the `Target` class corresponds to the target the subject intends to select while `Non-Target` designates all the other stimuli. For the training and copy-spelling phase, the targets were already known. For the free-spelling data, we added this information later on.

6 Additional information contained in the `data` structs

- `y_stim`: The stimuli consisted of six different geometric shapes with unique colors. This field specifies which of the stimuli was shown.
- `y_trialIdx`: As a trial, we denote the complete letter selection sequence including both levels, as shown in figure 1. In other words, one trial corresponds to one selected letter. This field contains the index of the trial an event belongs to. It can be used to e.g. select specific letter selections or investigate the effect of temporal location of a stimulus within a sequence.

7 Additional information contained in the `bbci_mrk` structs

The `bbci_mrk` struct has a number of fields that contain additional information about the experiment:

- `modeName`: Strings labelling the mode or the phase of the experiment, namely 'calibration' (initial phase collecting training data), 'copyspelling' and 'freespelling'.

- `nRepetitions`: The number of blocks or sequences. Should be 10.
- `letter_matrix`: a character matrix. Each row specifies the letters or symbols for each of the 6 letter groups.
- `desiredPhrase`: The phrase that the subject intended to spell in each of the three spelling phases (calibration, copy-spelling, free-spelling).
- `spelledPhrase`: The phrase that was actually spelled. For calibration, there is no phrase since no online spelling was performed. For copy-spelling, it should be 'LET_YOUR_BRAIN_TALK' for all subjects. For free-spelling, each subject chose his or her own phrase.
- `spelledSymbols`: The `spelledPhrase` only contains the final text after the subject was finished spelling. `spelledSymbols` contains also the misspelled letters and the backspace `'<'` that might have been used to correct mistakes. If the BCI system did not make any mistakes, `spelledSymbols` and `spelledPhrase` are identical.
- `mrk.event`

  The event field contains additional information for each of the events. In the present dataset, the following additional information is most important:

  - `trial_idx`: Same as `data.y_trialIdx`
  - `block_idx`: The stimulus sequence comprised 60 symbols that were flashed, subdivided into 10 blocks. Within each block, each of the 6 possible targets was shown in random order. Therefore, each of the 6 targets was shown 10 times. `block_idx` is a number between 1 and 10. If it takes the value n, it means that the corresponding symbol is shown for the $n-th$ time within the current stimulus sequence.
  - `idx_in_block`: Within each block (sequence), each of the 6 symbols was shown in a random order. `idx_in_block` is a number between 1 and 6 that gives the position of the stimulus within one block.
  - `stimulus`: Same as `data.y_stim`
  - `mode`: a matrix of size [number of stimuli $\times$ 3]. Each row has two 0's and exactly one 1 indicating whether a stimulus belongs to the 'calibration' phase (column 1), the 'copyspelling' phase (column 2), or the 'freespelling' phase (column 3). Type `sum(mrk.event.mode)` to print the amount of events in each phase.

  For the following substructures, the onset information in samples is contained in the `pos` field. To convert into milliseconds, use `pos/data.fs * 1000`.
- `mrk.misc`: Miscellaneous events.

- `run_start`: Onset of a calibration, copy-spelling, or free-spelling run. For some subjects, some of these triggers are missing.
- `run_end`: End of run.
- `countdown_start`: Start of the countdown before level 1 of the letter selection.
- `end_level1`: End of level 1, the group selection stage (upper part of Figure 1).
- `end_level1`: End of level 2, the letter selection stage (lower part of Figure 1).
- `invalid`: During the stimulation, the eye movements of the subject were monitored using an eyetracker. If the subject did not fixate, the trial was re-started and an 'invalid' trigger was recorded.

8 Selecting events using the `bbci_mrk` struct

8.1 Selecting target presentations for a particular iteration block of the stimulation

The stimulation sequence (showing each of the 6 targets in a random order) was repeated 10 times, so that there were 10 blocks. To select the stimuli belonging to a particular block the variable `mrk.event.block_idx` can be used. For instance, to select only trials in the 5-th block, use: `block5 = (mrk.event.block_idx == 5); data.trial = data.trial(block5); data.y = data.y(block5);`

References

[1] M S Treder, N M Schmidt and B Blankertz Gaze-independent brain–computer interfaces based on covert attention and feature attention. J Neural Eng, 8:6, 2011.