Experimental Design: Feature Attention BCI

The purpose of this assignment is to get you to build a complete working BCI experiment with working stimulus and signal-processing to make all 3 phases of a normal BCI task, i.e. Calibration, Classifier Training, and On-line feedback.

This document describes in detail the BCI you should build and how it should work for each of the stages. Based on the work you have done during the tutorial stages of the course you should be able to complete this task in about 5hr of work.

Note: This is an individual project, so you should write your solution on your own. Discussions of solutions are of course fine, but write all the code yourself.

Grading

This assignment will be worth 10% of the final grade

- The assignment is graded as 3 sub-parts over 3 weeks.
- Each weekly assignment will be graded using two criteria:
- **Does it work?** 3/4 of the grade for this assignment (2.5% / week) will be based on whether the developed code works as required by the assignment description. This means that (a) it runs, (b) the stimuli have the correct properties, (c) the required events are sent to make it work as part of a BCI
  - **Note: This is a 0/1 grade. To pass the course you must get a passing grade for does-it-work on all 3 weekly assignments.**
- **Is it well written?** The final ¼ of the grade (i.e. 2.5% in total) is based on whether the developed code is clearly written, with sensible structure, variable-names, event-types/values, good comments. etc.

Submission

To complete this assignment you should upload 5 files:

1. **Part 1: Calibration-Stimulus.** (1 file)

   Upload a single file called **CalibrationStimulus.m** or **CalibrationStimulus.py.** Which generates the required stimuli and sends the appropriate annotation events.

2. **Part 2: Calibration-Signals + Training-Signals** (2 files)
   CalibrationSignals.m/.py -- a file which does the signal processing associated with the calibration phase
   TrainingSignals.m/.py -- a file which contains the code for training the classifier based on the saved calibration data

3. **Part 3: Feedback-Stimulus + Feedback-Signals** (2 files)
   FeedbackStimulus.m/.py -- a file which shows the stimulus and feedback for the feedback phase of the experiment
   FeedbackSignals.m/.py -- a file which contains the code for applying the trained classifier during the feedback stage.

N.B. You can assume this code will be tested from a sub-directory of the ***buffer_bci/tutorial*** directory. Thus the standard setup code for paths and initial buffer connection will run as normal.

Resits

You must pass this component of the course to pass the course. Also it is very important for you to have the practical skills tested in this assignment before proceeding to the main assignment phase of the course. Thus, there is a single resit opportunity for this component of the course with a deadline 2 weeks after the start of the main assignment phase, i.e. mid-November (check the assignment on **blackboard** for the exact date.) In this resit assignment you may re-submit any/all weekly assignments that you have failed in a single submission.

Experimental Design : Feature Attention BCI (Rapid Serial Visual Presentation)

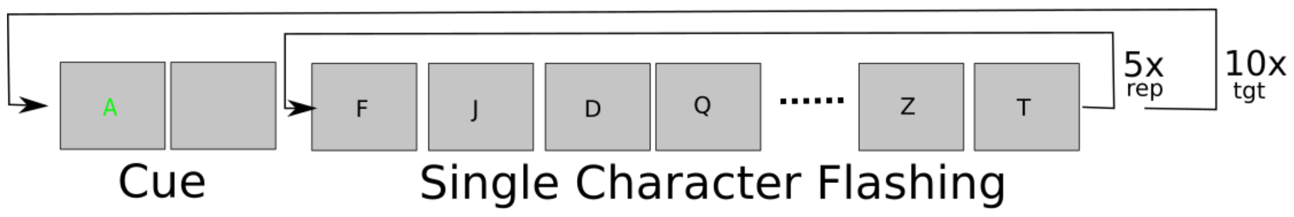
Introduction

A P300 or 'oddball' response is evoked whenever a target stimulus occurs in a stream of standard stimuli. What makes the 'oddball' particularly interesting for BCI purposes is that the definition of a 'standard' and 'oddball' is something the user can choose. Thus, the user can choose, for example to only treat the letter 'A' as a target and ignore all the other letters to enable a simple spelling BCI. This approach is exploited in the so-called Rapid Serial Visual Presentation (RSVP) BCI, to for example, identify target images in an image stream [1], selecting characters in an alphabet [2] or reconstruct a target shape from visual primitives [3].

Your task here is to build a simple RSVP based BCI for selection of letters in a stimulus stream, similar to that in [2].

Stimulus

The basic design of the stimulus is shown in the below figure.



Single Character Flashing

The basic stimulus presentation for detection of a single character consists of:

1. For 5 Repetitions : where a repetition is once through the alphabet (A..Z all 26 upper-case letters).
2. For each letter of the alphabet, where the order of letters is **randomized,**
3. show the letter for 100ms

Calibration Phase

During the calibration phase you need to present 10 target letters with target cues to the subject to get training data for the classifier. Specifically:

1. For 10 target letters, where the order of targets is **randomized**
2. Show a Cue with the target letter on the screen in 'Green' for 2 seconds
3. Clear the screen for 1 second
4. Run the normal single character stimulus process described above, i.e. 5 repetitions of all characters in randomized order.

N.B. Please use the file name **CalibrationStimulis.m/.py** for this component

Feedback Phase

During the testing phase you do not give a target letter but let the user choose for themselves. You do however need to present feedback on the target detected by the BCI. Specifically:

1 For 10 **testing** letters:

2 Show an instruction on the screen saying 'Think of your target letter and get-ready' for 2 seconds

3 Clear the screen for 1 second

4 Run the normal single character stimulus process described above, i.e. 5 repetitions of all characters in randomized order

5 Clear the screen for 1 second

6 Show the predicted target letter on screen in blue for 2 seconds.

Note: to show the predicted target letter you will need to 'decode' it from the sequence of stimuli used and the sequence of classifier predictions.

N.B. Please use the filename **FeedbackStimulis.m/.py** for this component

Signal Analysis

This is an evoked response BCI, so you will need to train an ERP classifier. Further the two classes of brain response expected are:

1. Target responses: which occur then the letter presented on screen is the users current target, i.e. the one they were cued with in the calibration phase, or the self selected target in the testing phase.
2. Non-target responses: which occur when **any other letter** than the target is presented on screen.

The exact signal we are looking for is a P300 response. So you will need to set the analysis parameters, (such as trial-duration, frequency filter parameters) for the ERP classifier to values appropriate to detect this type of response.

As always with a BCI, you will need a signal-analysis scripts for each experiment phase to:

1. Catch the stimulus events and data and save the labelled data during the calibration phase. (**CalibrationSignals.m/.py**)
2. Train a classifier with appropriate parameters based on the saved labelled data, and save this classifier. (**TrainingSignals.m/.py**)
3. Apply the trained classifier to the live data during the testing phase. (**FeedbackSignals.m/.py)**

Note: As noted above you will also need some decoding process to generate a predicted letter based on the classifier outputs and knowledge of the stimulus sequences.

References

[1] Gerson, Adam D, Lucas C Parra, and Paul Sajda. “Cortically Coupled Computer Vision for Rapid Image Search.” *IEEE Transactions on Neural Systems and Rehabilitation Engineering: A Publication of the IEEE Engineering in Medicine and Biology Society* 14, no. 2 (June 2006): 174–79. doi:10.1109/TNSRE.2006.875550.
[2] Acqualagnav, L., M. S Treder, M. Schreuder, and B. Blankertz. “A Novel Brain-Computer Interface Based on the Rapid Serial Visual Presentation Paradigm.” In *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2686–89. IEEE, 2010. doi:10.1109/IEMBS.2010.5626548.
[3] Seoane, Luís F., Stephan Gabler, and Benjamin Blankertz. “Images from the Mind: BCI Image Evolution Based on Rapid Serial Visual Presentation of Polygon Primitives.” *Brain-Computer Interfaces* 2, no. 1 (January 2, 2015): 40–56. doi:10.1080/2326263X.2015.1060819.