Flinders UNIVERSITY
ADELAIDE · SOUTH AUSTRALIA

Towards vibrotactile mobility control using the P300 in a BCI

By

Tuan Anh Vuong

Thesis
Submitted to Flinders University
for the degree of

Master of Engineering (Biomedical)

College of Science and Engineering
31/05/2024

TABLE OF CONTENTS

ABSTRACT

With the brain-computer interface (BCI), besides using visual and auditory stimuli, the user could control the device using vibrotactile stimuli to utilise the P300 event-related potential. However, the effect of the location of vibrotactile stimuli on the body and its performance is unclear. This project investigated the correlation between the location of the tactors on the body and the accuracy of the participant in recognition of the target tactor. From there, recommendations for tactor location setup were derived for optimal extraction of P300. In the experiments, 4 tactors were used and set up in 5 configurations of tactor location. The EEG and EMG signals were recorded in each configuration using the EEG scalp and the EMG electrodes. The EMG signal was used to grade the participants' responses' accuracy. The P300 ERP analysis was performed to extract and average the EEG signal from the 300 to 500 ms range. The decision matrices built from the EMG data were used to validate the accuracy, precision and recall of each configuration of tactor locations. The project results were able to show the accuracy in focusing on the target tactor based on the locations through 5 tactor configurations. Then, the project provided recommendations for potential locations of vibrotactile stimuli setup for effective P300 ERP extract with respect to one arm, both arms and under regions with dense nerve fibre. The results are expected to provide a better understanding of the vibrotactile stimuli, as well as improve the paradigm for extraction and classification of the P300 ERP waves in BCI applications. The findings could be extended for further investigation, which includes how the number of tactors affects the performance of the participant, as well as how the tactor configurations jeopardise the participant’s performance on a larger scale with multiple participants.

DECLARATION

I certify that this thesis:

1. does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any university
2. and the research within will not be submitted for any other future degree or diploma without the permission of Flinders University; and
3. to the best of my knowledge and belief, does not contain any material previously published or written by another person except where due reference is made in the text.

Signature of student.....

Print name of student.....Tuan Anh Vuong.....

Date.....31/05/2024.....

I certify that I have read this thesis. In my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Engineering (Biomedical). Furthermore, I confirm that I have provided feedback on this thesis and the student has implemented it entirely.

Signature of Principal Supervisor.....

Print name of Principal Supervisor.....Kenneth Pope.....

Date.....31/05/2024.....

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to Associate Professor Kenneth Pope, who has been an incredible supervisor and mentor, not only throughout my Master's Thesis Project but also during my Master's in Biomedical Engineering journey at Flinders University. Your support and feedback have been extremely valuable to me, which not only helped make my thesis journey more enjoyable but also helped me to learn a lot about how to approach and analyse problems. I am impressed with how you think when writing the Matlab code and analysing the data. I will take your coding style with me and try my best to adapt it.

Thank you, Eric Deng, for being my participant in my project. Your contribution helped my project move through the critical period. I enjoyed our chat time, especially about the mechanical keyboard. And I will remember the time I got to play with your head during the EEG setup.

Thank you, Lynton Graetz and Bek O'Loughlin, for your helpful advice and support during my experiment protocol and set-up. Thank you for sharing your experiment of your research journey with me. I was able to push the project forward thanks to your support.

Lastly, I would like to thank my parents. Their belief and support have always been a huge support that helped me keep moving forward. Nothing I have til now will be possible without them.

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

Besides brain disease diagnostics and brain behaviour studies, controlling and interacting with the surrounding objects using the electroencephalogram (EEG), such as a spelling machine for speech impairments, drone control, home application control, and wheelchair driving for disability people using EEG are exciting applications that have interested scientists and engineers.

Such applications utilise the human event-related potential (ERP), which is the positive and negative peaks in the potential of the brain that occur with stimuli (Sur and Sinha, 2009). ERP is used to extract the brainwaves, which are helpful for identifying the intention of the user in communicating with the environment, such as driving a wheelchair or moving a cursor on the computer. ERP for interacting with the external world is the Brain-computer interface (BCI) (Ortner et al., 2021). In the human event-related potential, the P300 wave, which is the wave that occurs in the EEG signal when the participant experiences sudden stimuli (Hajian and Yong, 2017), results in a sudden increase in the EEG signal, with a delay of around 300 ms after the stimuli. The extraction of the P300 wave to obtain the ERP is a common method. The stimuli could be visual, auditory and vibrotactile.

The application of mechanical vibration to the skin surface generates the vibrotactile feedback. This vibration contains information on vibration frequency, amplitude, duration and spatial location (Fleury et al., 2020). This will significantly help people with visual and auditory impairment, where the visual and auditory stimuli are not applicable. Previous works (Belhaouari et al., 2018, Duan et al., 2023, Kono and Rutkowski, 2015, Mori et al., 2013) have shown vibrotactile stimuli's potential in eliciting the P300 ERP. Investigation into the feasibility of using vibrotactile stimuli in eliciting the P300 ERP, which helps identify the user’s intention using their attention to vibrotactile stimuli, can be helpful in mobility control applications since the vibrotactile paradigm provides the ability for the user to look and hear while still being able to control; for example, a wheelchair, as they do not need to focus on the screen to control.

One device that delivers mechanical vibration to the skin is the tactor, the actuator that creates the vibration stimuli to the skin. An example of a tactor is the vibration haptic on a console controller such as PlayStation. Previous works have investigated the efficiency of tactors in delivering vibrotactile stimuli. Mori et al. placed the tactile in ten fingertips to evoke the somatosensory brain response, which could be applied in robotic vehicle operation (Mori

et al., 2013). However, only three commands per minute were achieved, and the tactile was only performed at the fingertip. Another study utilised the motor imagery paradigm to control a virtual cursor with only vibrotactile biofeedback (Chatterjee et al., 2007). The subjects were able to control the BCI with up to 72% accuracy. The study also found that the vibrotactile feedback placement could affect the BCI accuracy; however, only biceps ipsilateral or contralateral were investigated. Belhaouari et al. performed the BCI using vibrotactile stimuli on the face and fingers (Belhaouari et al., 2018). The study showed that the clear shape of P300 obtained was related to the proposed tactile BCI. These previous works showed promising results in utilising vibrotactile in the BCI to extract P300; however, they also lacked investigation into the effect of vibrotactile stimuli placement on the participant on the accuracy of the BCI.

The project hypothesis is that the locations of the tactors on the user's body could provide different performances in extracting the P300 wave from the user. The experiments were conducted so that participants would actively focus on the desired tactor and tap the index finger to respond when four tactors vibrate one after another. When the target tactor vibrated and the participant recognised it, the EEG recording system could record the P300 ERP wave of the participant, as well as the EMG signal from tapping finger. Therefore, the project aimed to investigate the effect of tactors' location on the body on the accuracy of choosing the desired tactor for the user. The findings could examine the correlation between the tactors' location and the ability to perceive the target tactors. Then, the project could recommend the location to better elicit P300. From these findings, further investigations into optimising the P300 eliciting using tactors for accessibility applications could be carried out. This thesis covered the Methodology, Results, Discussions, and Conclusion, as well as suggested possible Future Work from the project.

LITERATURE REVIEW

Brain-Computer Interface

The brain-computer interface, or BCI, is a system that allows humans to interact with the surrounding environment without the need for peripheral nerves and muscles. The potential in brain activities is the control signal. The BCI system records the brain signal, analyses it to understand and predict the user's intention, and then sends the command to external devices (Nicolas-Alonso and Gomez-Gil, 2012).

The classification of BCI systems depends on dependability (dependent or independent), invasiveness (invasive or non-invasive) and synchronisation (synchronous or asynchronous) (Ramadan and Vasilakos, 2017). Various neuroimaging methods exist to collect brain activities, including invasive and non-invasive methods. While the invasive methods could provide clean signals, there are risks related to surgical procedures. Therefore, non-invasive methods, such as electroencephalography (EEG), magnetoencephalography (MEG) and functional MRI (fMRI), are preferred for future commercial potential (Min et al., 2010).

Electroencephalography (EEG)

Electroencephalography, or EEG, measures the differences in the electrical potentials of the electrocellular activity of the cortical neurons using the electrodes at the scalp (Kaiser, 2005). The amplitude of a typical EEG signal is about 10 to 100 µV, with the frequency range from 1 Hz to 100 Hz. The popular electrode location is the international 10 – 20 system, which is based on the location of an electrode and the area of the cerebral cortex (Negi, 2021, Subha et al., 2010). The EEG has been a valuable tool in diagnosing and studying many conditions, such as sleep health and epilepsy, as well as psychological issues, such as meditation (Subha et al., 2010).

The EEG-based BCI system is the most common among the BCI systems thanks to its good temporal resolution, low risks and cost efficiency (Nicolas-Alonso and Gomez-Gil, 2012, Vaid et al., 2015). This advantage of EEG could help the researcher understand the operation of the brain in real-time. The system consists of an array of electrodes to obtain the signal of brain activities. After processing, the feature extraction and classification stages help transform these signals into commands. Therefore, this system allows people with motor or visual impairments to interact with the surrounding environment. BCI has shown

promising applications in health care, such as assistive technology (wheelchair, spelling machine), rehabilitation, and monitoring of mental and cognitive states (Zabcikova et al., 2022).

Event-Related Potentials (ERPs)

Event-related potentials, or ERPs, are minor changes in voltages in brain activities when the brain responds to specific stimuli or events. The changes are time-locked to sensory, motor, or cognitive events; hence, it is also called time-locked EEG activity. There are two categories: the early waves that peak within the first 100 milliseconds after the stimulus and the waves that peak after 100 milliseconds of the stimulus. Based on the positive and negative amplitude of the wave, as well as its latency after the stimulus, the ERP waveforms are named P (positive) or N (negative), with latency (Im, 2018, Sur and Sinha, 2009). In these ERP waveforms, P300 is the most popular waveform. P300 is the positive peak in brain waves with a latency of about 300 to 500 milliseconds after the stimuli (Hajian and Yong, 2017, Pérez-Vidal et al., 2018). It is elicited using an oddball paradigm when the brain detects an unusual stimulus (target stimulus) in a series of stimuli (Picton, 1992). The stimuli could be visual, auditory or somatosensory. Although various researchers utilised the auditory and visual stimuli thanks to their easy control, they might not be suitable for people with impaired visual or auditory.

Vibrotactile stimuli in BCI

Moreover, the user could experience fatigue caused by looking at a display for a long time or focusing on the audio signal (Huang et al., 2022, Mak et al., 2011, Saha et al., 2021). Therefore, various investigations have been conducted into the use of vibrotactile stimuli for BCI applications. The most popular vibrotactile stimuli are the delivery of mechanical vibration to the skin surface. The vibration could be configured in frequency, amplitude, duration and spatial location (Fleury et al., 2020). The human could also distinguish the difference in vibrating patterns (Kim et al., 2016), which is ideal for a patient with paralysis, while the vibro tactor could be hidden in clothes, which provides the user with the freedom to look around without having to pay much attention to a display or a sound.

Mori et al. investigated the feasibility of operating a robotic vehicle using the tactile stimuli at the fingertips. They achieved three commands per minute, but only the tactile at the fingertips was studied (Mori et al., 2013). Chatterjee et al. also studied the control of a virtual cursor using vibrotactile biofeedback. In this study, they found that there could be

biases in the placement of the vibrotactile stimuli to the accuracy of the system (Chatterjee et al., 2007). However, the study only investigated the placement at the biceps ipsilateral or contralateral. In another study, Belhaouari et al. studied the vibrotactile stimuli BCI system with the stimuli placement on the face and fingers (Belhaouari et al., 2018). The result showed that they could obtain a clear P300 signal. Ortner et al. investigated a BCI system to reduce the wrong answers, using 7 tactile simulators on different parts of the subject. The study was able to achieve an accuracy of up to 95%. However, the effect of tactile location on the accuracy required further study (Ortner et al., 2021).

Although these studies in the vibrotactile P300 BCI in various applications, with various tactile placements, have shown the feasibility and potential of the vibrotactile P300 BCI system, few studies investigate the bias of tactile stimuli placement to the accuracy of the vibrotactile BCI system. Therefore, the investigation in this project hopefully could provide a clearer picture of optimising the vibrotactile P300 BCI. This project aimed to figure out: whether P300 can be obtained with an oddball paradigm using 4 tactors, can the participant recognise which tactor vibrates better when they are further apart, and finally, recommendations of potential tactor setups for eliciting P300 in future wheelchair applications. Moreover, based on this investigation, further studies could be carried out, such as the effect of the number of vibrations and the impact of different vibration frequencies on the performance of the user.

METHODOLOGY

Options analysis

An analysis of options was performed to ensure the feasibility of the project and answer the research question derived from the literature review.

Hardware

The tactors used in the project were the Eccentric Rotating Mass (ERM) vibration motor in a coin shape due to the suitable compact size and vibration frequency. The tactors were assumed to function correctly, and the errors between tactors were minor and acceptable. The Raspberry Pi was the microcontroller (MCU) used to drive the tactors, which operated on Raspbian OS; hence, it was convenient to create the oddball paradigm with prompts displayed on the screen. For the EEG and EMG recording, g.Hiamps and the headbox from g.tec were available in the lab and provided up to 128 inputs, which was more than enough. The whole experiment took place inside the Multimodal Recording Facility's Faraday cage, which helped eliminate the 50 Hz electrical noises and other signal noises that could affect the EEG recording, which was very sensitive to noises. This project also assumed the Faraday cage functioned adequately.

Software

Python was the programming language used to control the RPi and deliver the oddball paradigm. The RPi operated on Raspbian OS, which already included Python, a lightweight and well-supported programming language. Moreover, Python had NumPy, a well-developed library that works with arrays and matrices, which was necessary for constructing the oddball paradigm. For the data recording and analysis, Matlab and the lab's standard toolbox, eeg3, were used. The eeg3 toolbox was developed and set up to provide a seamless workflow between the recording facility and data analysis.

Tactor location configurations

With the inspiration of the BCI wheelchair application, the tactor configuration should provide convenience in both design and user setup. Therefore, investigating having tactors attached to one arm with different location configurations and compare with having tactors attached to both arms could provide interesting insight into optimal setups. The participant was right-handed, and the tactors were set up on the non-dominant hand (left hand), with

the exception of Trial 4, where both arms were used. Figure 1 shows all configurations used in the experiment.

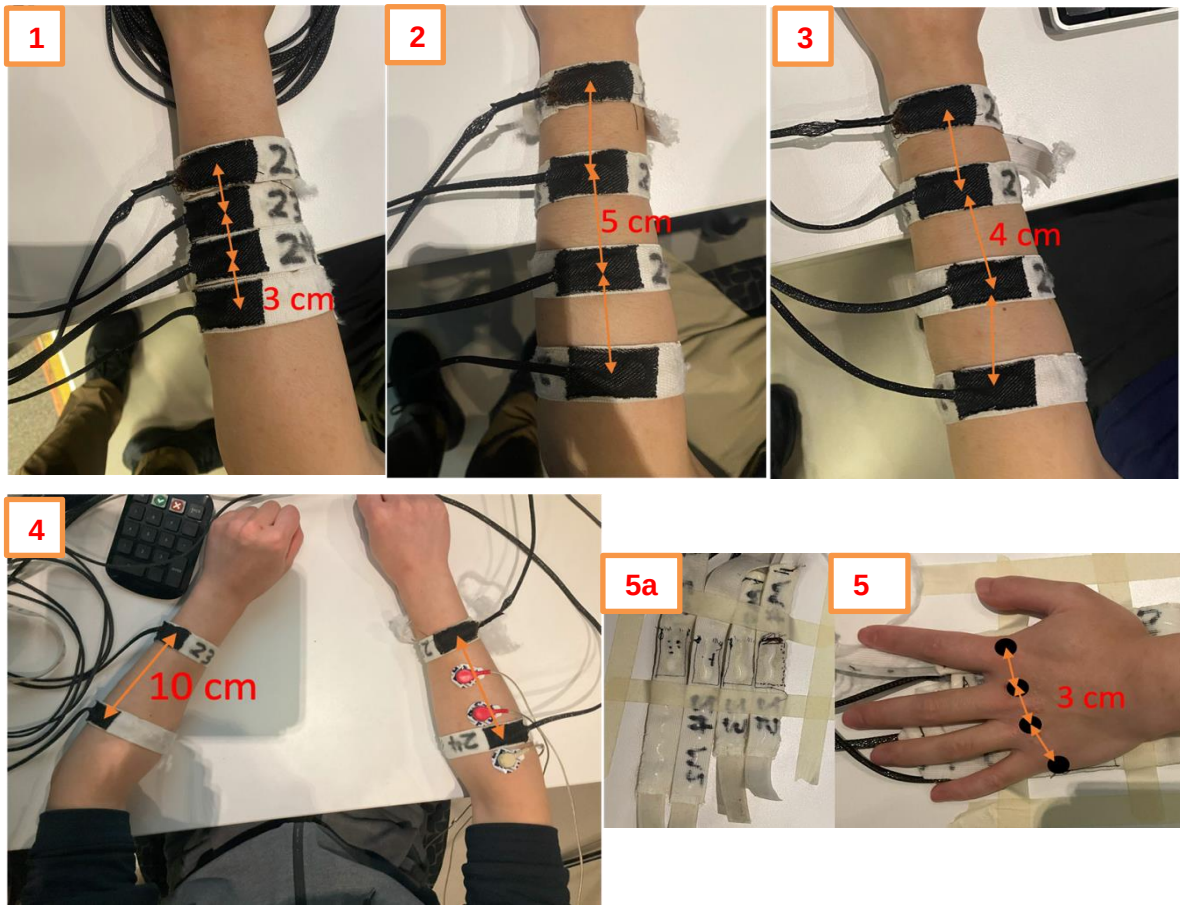

Figure 1 – Five tactor configurations in 5 trials. 1) Trial 1: left arm, spacing between tactors: 3 cm. 2) Trial 2: left arm, spacing between tactors: 5 cm. 3) Trial 3: left arm, spacing between tactors 4 cm. 4) Trial 4: two tactors on each arm, spacing between tactors: 10 cm. 5a) Trial 5's tactor setup. 5b) Trial 5: left-hand palm, spacing between tactors: 3 cm.

Oddball paradigm

To elicit the P300, the participant needs to be involved in the task. Hence, the paradigm should be designed in a way that is challenging enough but not too challenging or too easy. P300 waves occur when the participant experiences infrequent stimuli. Therefore, the oddball paradigm should deliver the target stimuli less frequently than non-target stimuli. Since there were 4 tactors used in this project, the ratio of 25/75 was sufficient (all tactors had a 25% vibrating chance). The order of target and non-target stimuli was random in a way that the first 5 stimuli would never be the target stimuli to eliminate the surprise factor, and there were no 2 consecutive target stimuli in a row to minimise false positives.

One trial corresponded to one tactor configuration mentioned above. Each block in one trial corresponded to a specific target tactor. The vibration of each tactor was about 250

ms, with the interval between 2 vibrations randomly in the range of 0.9 to 1.4. With 40 stimuli per block, completing one block would take about 2 minutes. There were 4 tactors, so each tactor needed to be a target. Since with 4 blocks, the paradigm would be too short, as the ERP, 8 blocks with each tactor being the target twice would be sufficient. In total, one trial with 8 blocks and 40 stimuli per block could take around 25 to 30 minutes, plus rest time, practice time and introduction on the prompts.

EEG and EMG montage

An author investigated EEG montage configurations with 8, 16, 32, and 64 electrodes and found that the 16-electrode configuration could provide a similar result with 32 or 64 electrodes (Hoffmann et al., 2008). The most important electrodes were the midline ones, from AFz to Oz. Therefore, the EEG montage used in the project followed this recommendation with some extra electrodes for potential interesting findings. In total, there were 22 EEG electrodes: AFz, Fz, FCz, Cz, CPz, Pz, POz, Oz, FC1, FC2, C1, C3, C2, C4, CP1, CP2, P3, P7, P4, P8, O1, O2. The reference electrode was FCz.

Two EMG channels provided the muscle signals when the participant taped the index finger, which helped identify whether the participant recognised the correct stimulus. To minimise the noise, three EMG sensors were used: two sensors for recording the difference in potential and one sensor for reference.

Participant assumptions

The participant was assumed to have normal muscle contraction, so the EMG signal of the participant is normal to distinguish with the vibration of the tactors via EMG recording since, in the configuration of Trial 4, tactors were placed near the EMG sensors. Also, the participant was assumed to have normal brain activity, which meant their EEG signals were normal, with minimum noise and artefacts. The neurological function of the participant was also assumed to be normal for optimal results.

Experiment pipeline

Each trial in the experiment was conducted following the same pipeline. The participant's head and arm were cleaned with alcohol scrubs to reduce the impedance of the electrodes. The EEG cap with pre-attached EEG electrodes was put on, and each electrode was filled with gel to increase the contact between the scalp and the electrode. The impedance was checked on Matlab to ensure the values were less than 10 kΩ. Tactors

were placed on the arm according to the configuration. EMG electrodes were placed on the dominant hand, as shown in Figure 2. The participant was required to tap the index finger when he felt the target stimuli. During the experiment, the Raspberry Pi drove the tactors according to the oddball paradigm and displayed the prompts for the participant. The participant was allowed to do multiple training blocks to get familiar with them before starting the actual block of the experiment.

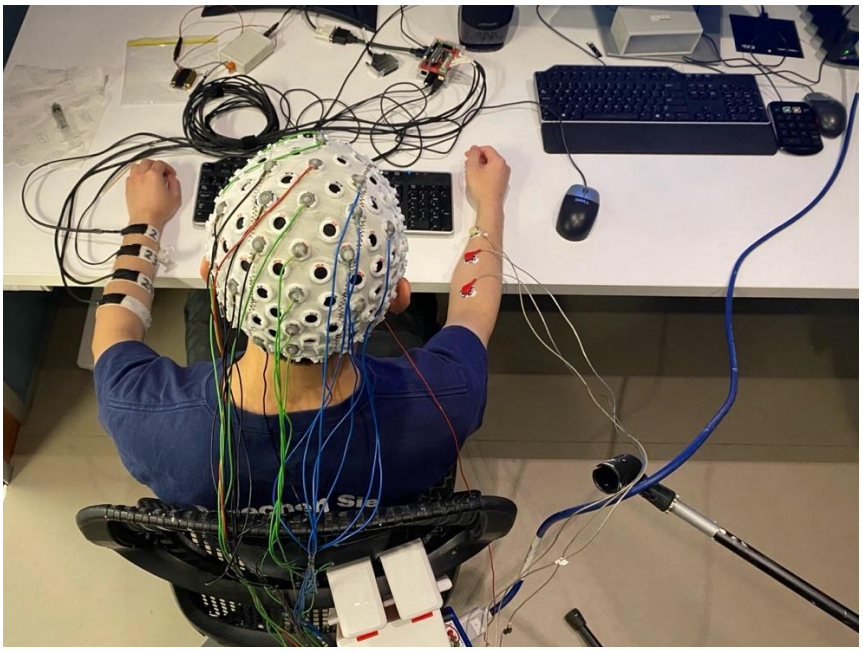

Figure 2 – EEG experiments setup. Tactors were attached to the left arm. EMG sensors were attached to the right arm. EEG cap with 22 EEG channels.

Data analysis

The recorded data was imported and analysed in Matlab scripts using the eeg3 toolbox. For filtering (Zhang et al., 2024), the recommended settings for high highpass and lowpass filters applied for the P3 component were a 0.2 Hz cutoff for the highpass filter and a higher 10 Hz cutoff for the lowpass filter for easier visualisation of the P300. Therefore, the cutoff frequency for the highpass filter was 0.2 Hz and for the lowpass filter, 25 Hz. The offset, or the measurement window, was set to -0.3 to 0.7 to investigate not only the 300 - 500 ms range of P300 but also the behaviour of signals before and after that. In the recorded EEG time series, the tactor events were labelled as "1", "2", "4", and "8", corresponding to tactor 1, 2, 3, 4. At the beginning and at the end of each block, all four tactors vibrated in pre-determined orders and served as the block marker. Each order corresponded to a specific block (i.e. 1248 for practice blocks, 8214 for the start of block 1, 2184 for the end of block 1 ...). These pre-determined orders were recoded into unique numbers that

corresponded to specific blocks for extracting purposes. Then, EEG epochs were extracted into each block according to the recode events. At the beginning of each block, the participant was allowed to repeatedly vibrate the target tactor to get familiar. Therefore, the initial target tactor events created by the participant in the recording were removed from the event list in each epoch. After cleaning the data and extracting epochs, the target and non-target responses were averaged across each trial in the offset range (-0.3 to 0.7). As recommended, the baseline correction was performed from -200 to 0 ms (Zhang et al., 2024). The means of these responses were plotted to observe the difference between target and non-target responses. The difference between target and non-target responses was calculated and plotted to visualise whether the increase in space between tactors was proportional to the difference between target and non-target responses. For the statistic test, a t-test was performed and visualised as a patch in the region where the difference was significant.

In each EEG epoch, pulses in EMG responses could be observed when the participant taped the index finger, which showed as pulses in the recording shown in Figure 3. The EMG pulses in each block were counted to construct the confusion matrix, which determined the accuracy of the participant's responses.

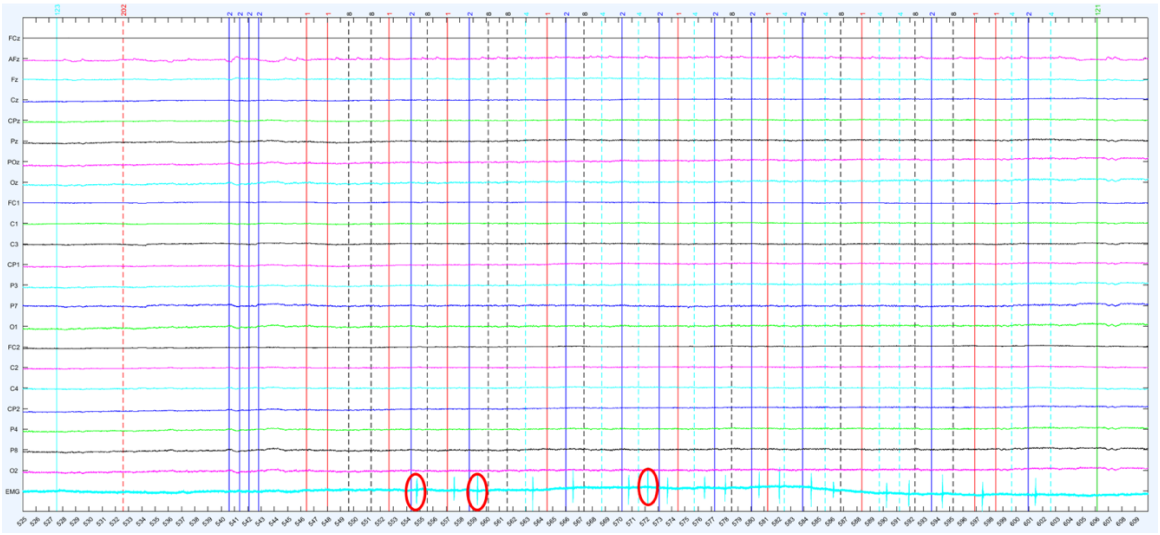

Figure 3 – An EEG epoch with an EMG channel showed pulses (red circle) due to index finger tapping.

Sources of experimental error

One of the error sources was the muscle due to the participant moving his dominant hand (the hand with EMG sensors) to press the keyboard while doing the index finger tapping task. These movements could cause some fluctuation in the EMG recording. To minimise, a small numpad, instead of a full-size keyboard, was placed right next to the

participant's dominant hand to minimise the movement. The other error source was the head muscles, which were minimised using highpass and lowpass filters. However, over-filtering could distort the signal. Another source of noise was the 50/60 Hz noises from AC sources, which were minimised using a Faraday cage, assuming that it functioned normally. The participant might feel stressed, not relaxed, and bored during the experiment. To minimise these effects, the experiments were conducted at the time that the participant felt most alert. The participant also chose the date he felt comfortable and did not have to deal with other commitments.

RESULTS

The experiments were conducted from 2 April to 19 April inside a Faraday cage at the Multimodal Recording Facility in Tonsley between 10:30 a.m. and 12:30 a.m.

P300 with the oddball paradigm using 4 tactors

After performing epoch extraction and averaging the response of target and non-target around the tactor events, each EEG channel in the montage was observed. From the plots in Figure 8A to Figure 14A , the midline channels AFz, Fz, Cz, CPz, Pz, POz, and Oz show distinct changes in the difference in response between target and non-target signals. For the P300, the midline channels Cz, CPz, Pz, POz, and Oz successfully show the P300 component, but the Pz channel is the easiest to observe.

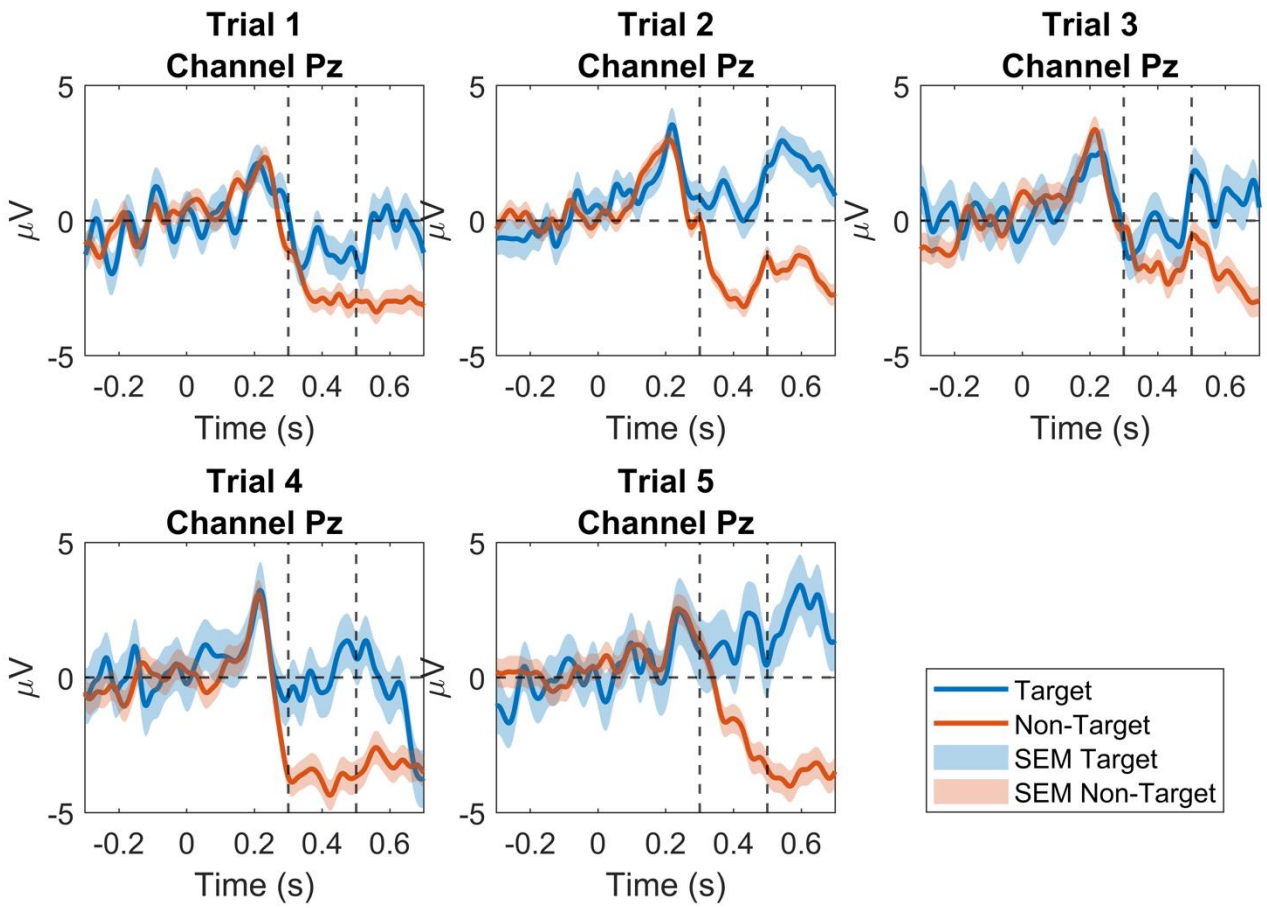

Figure 4 – Target and Non-Target responses and their SEM patch in each trial of the Pz channel.

Each trial had 8 blocks, with 40 stimuli in each block. In order to test the repeatability of the result, the standard error of the mean (SEM) was calculated and plotted as transparent

patches around the response mean lines. Figure 4 shows the SEM of target and non-target responses in all 5 trials of channel Pz. The SEM transparent patch has a similar shape to the mean lines of target response and non-target response. The upper and lower bounds of the patch are close to the mean lines, which indicates that the results in this project were repeatable.

Tactor locations and EEG amplitude of target responses

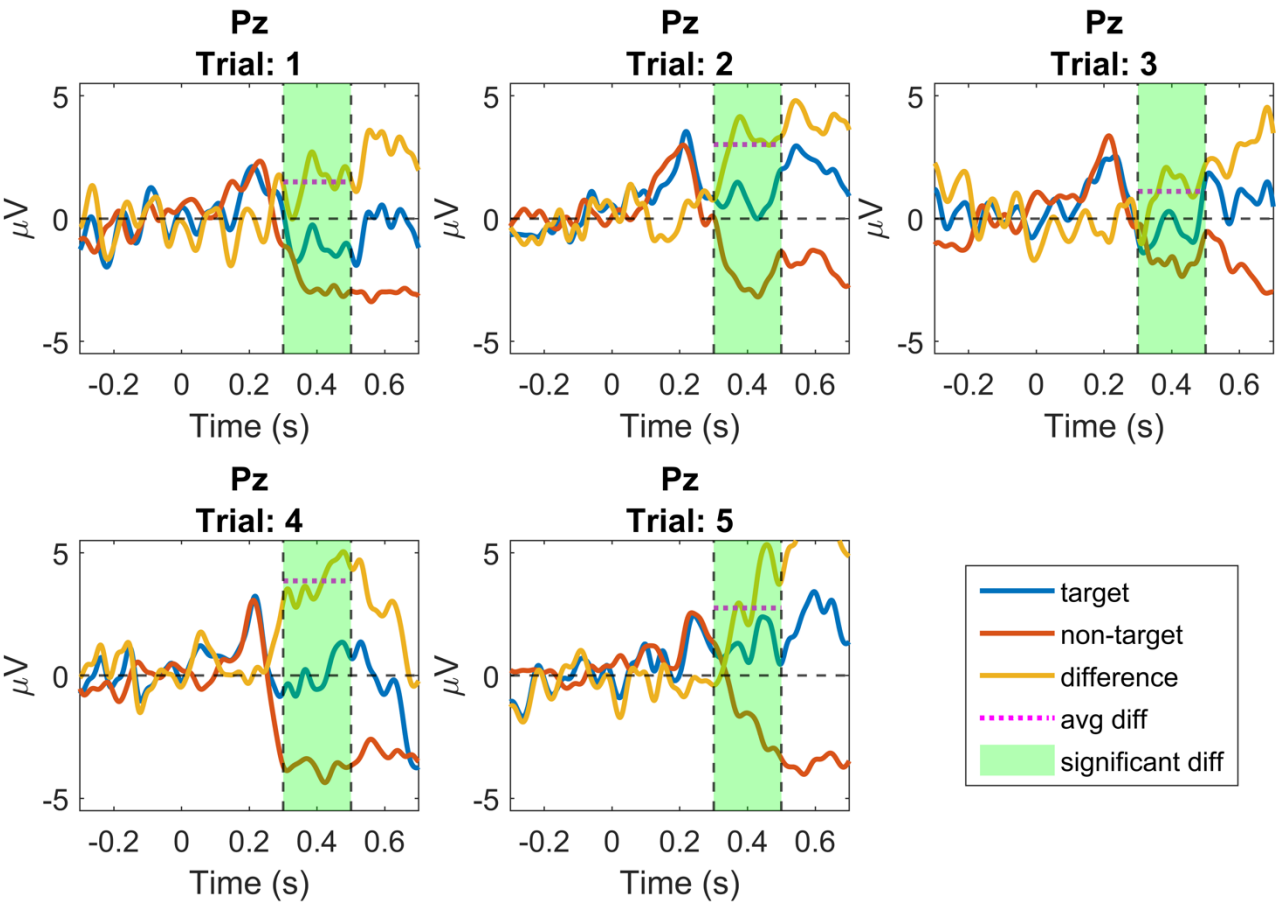

Figure 5 – Target, Non-Target responses, their difference line and green patch shows significant differences between Target and Non-Target responses in t-test, in each trial of Pz channel.

Figure 5 shows the target and non-target responses of 5 trials (each trial corresponding to one tactors configuration) in the Pz channel. The peak of the P300 component could be observed near 400 ms in Trial 2 and slightly later than 450 ms in Trial 4 and Trial 5. Trial 1 and Trial 3 show no P300 peak in the 300 to 500 ms range. The difference between target and non-target response (yellow line) increases as the distance between tactor increases. The average line in the range from 300 to 500 ms (magenta line) shows a slight difference between Trial 1 and Trial 3. Trial 2 and Trial 4 show a slightly large difference between each other and a significant difference compared to Trial 1 and Trial 2.

A t-test performed in the 300 to 500 ms range showed significant differences between the two responses in all 5 trials, which are indicated as the green patch in the Figure.

Figure 6 – Target responses in all 5 trials, showing the size of the P300 response between 300 and 500 ms, and the increase in amplitude as the spacing between tactors increased.

Figure 6 shows the target responses of all 5 trials in one graph. In the range from 300 to 500 ms, the amplitude of the target responses increased as the spacing between tactors increased. In Trial 1 and Trial 3, the spacing between tactors was quite similar (3 cm and 4 cm), which were very close to each other; the amplitude in the 300 - 500 ms range was small. On the other hand, Trial 2 increased the spacing to 5 cm, and the amplitude rapidly increased. This behavior could be observed in Trial 4. Trial 5 used the same spacing between tactors as Trial 1, but placed under the palm of the participant's left hand instead of the left arm, had a higher amplitude in the 300 - 500 ms range when compared to Trial 1.

Tactor locations and the difference between target and non-target responses

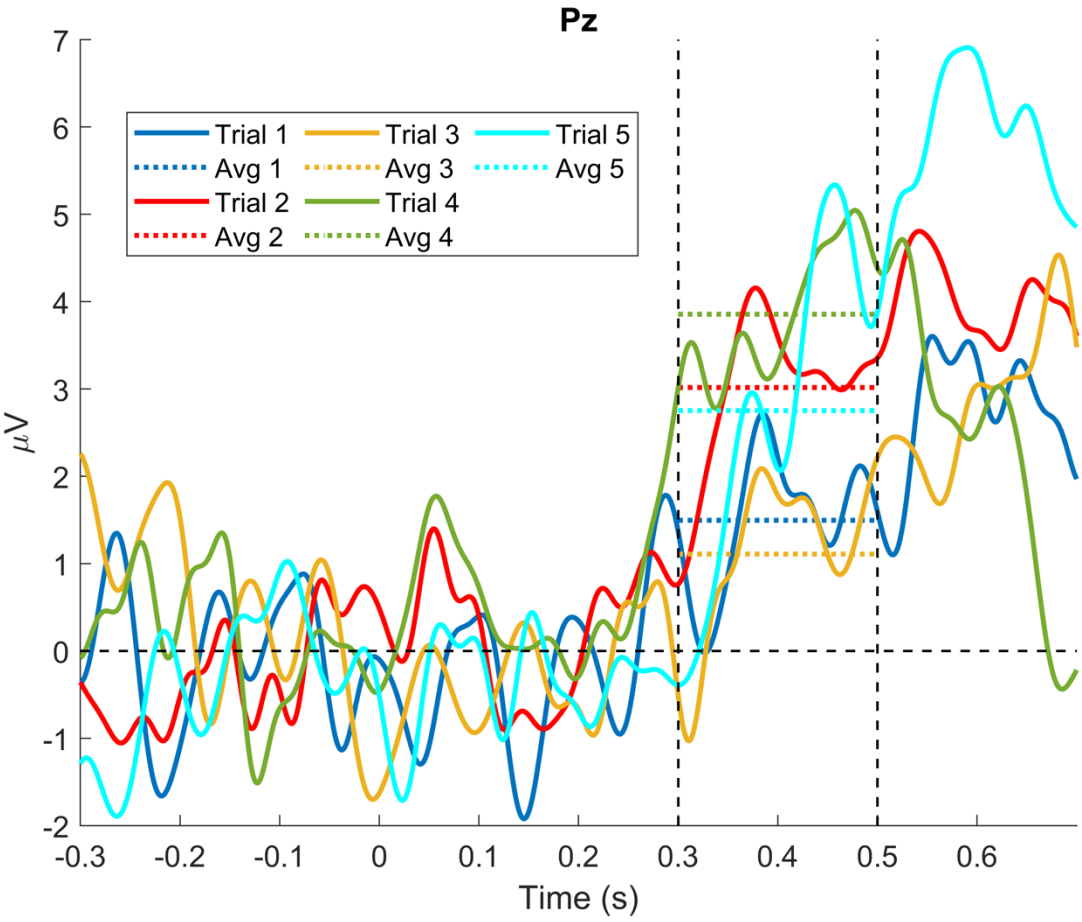

Figure 7 – Difference between the averaged target and averaged non-target responses for the 5 different trials, and the average of the waveforms between 300 and 500 ms, showing the size of the P300 response.

Figure 7 shows the difference line of target and non-target responses in all 5 trials in channels Pz. In the range from 300 to 500 ms, the difference line had a tendency to increase through each trial. The mean of each difference line from 300 to 500 ms was plotted. The mean lines of Trial 1 and Trial 3 are pretty close to each other. The mean lines of Trial 2 and Trial 5 are also close to each other and much higher than Trial 1 and Trial 3. Trial 4 showed the highest mean value of the difference line between 5 trials. The mean line of Trial 5 was much higher than the mean line of Trial 1, while the spacing between tactors was similar.

Tactor locations and Accuracy, Recall, Precision

		Participant									
		Trial 1 (3 cm)		Trial 3 (4 cm)		Trial 2 (5 cm)		Trial 4 (10 cm)		Trial 5 (3 cm, palm)	
	320	Target	Non-Target	Target	Non-Target	Target	Non-Target	Target	Non-Target	Target	Non-Target
Actual	Target	62	18	64	16	69	11	77	3	73	7
	Non-Target	48	192	37	203	40	200	0	240	15	225
Recall		0.78		0.80		0.86		0.96		0.91	
Precision		0.56		0.63		0.63		1.00		0.83	
Accuracy		0.79		0.83		0.84		0.99		0.93	

Table 1 – Confusion Matrix with Recall, Precision and Accuracy table.

The EMG pulses observed in each EEG epoch of each block derived the number of correct and non-correct reactions of the participant to the right stimulus. A confusion matrix throughout 5 trials was constructed to calculate Recall, Precision and Accuracy. According to Table 1, as the distance between tactors increased, the Recall, Precision and Accuracy also increased. Moreover, Trial 5 provided quite similar outcomes as Trial 4 while using the same spacing between tactors as Trial 1, with the only exception that the tactors were placed under the palm of the participant's left hand instead of on the left arm.

DISCUSSION

P300 elicit using oddball paradigm with 4 tactors

The results showed that the oddball paradigm used in this project was able to elicit the P300, which could be observed in Pz, CPz, POz, and Oz. However, Pz showed better potential since this channel provided both a clear P300 waveform and usability. The Pz channel is easier to use for EEG electrode setup than the POz of Oz channels, which are more at the back of the head and sensitive to muscle noises from the neck. Since BCI applications aim for usability, less electrode setup is a big advantage.

One interesting note is that the peak in amplitude of the target responses in the 300 - 500 ms range in Trial 4 has a longer latency when compared to Trial 2. The challenge of the Trial 4 could be a reason since this trial had 2 tactors in each arm with a spacing of 10 cm on each side. The participant confirmed this setup was easy and not challenging. The P300 wave also depends on the level of involvement of participants in the task; too easy or too challenging would affect the P300 elicit.

Effect of tactor locations on the ability to recognise the target stimuli

The results also showed that the difference in response between target and non-target signals increased as the spacing between the tactors on the arm increased. The closer the tactors, the harder it is for the participant to distinguish between the target and non-target stimuli. Trial 1 and Trial 3 had similar effects on target response amplitude in the 300 - 500 ms range due to the spacing between tactors being too close. In Trial 2, tactors were spread evenly as far as possible, according to the participant's left arm length, in which the target response started showing a P300 wave. Although Trial 4 was not as challenging as the other trials, this configuration could also elicit a P300 wave. The Accuracy, Precision and Recall derived from the Confusion Matrix of EMG data also provided evidence that the participant could tell which tactor was vibrating better as they were further apart.

One interesting finding was the huge difference between Trial 1 and Trial 5, given that these trials share the same spacing between tactors, with the only difference between tactors on the left arm (Trial 1) and under the left-hand palm (Trial 5). Trial 5 was able to elicit P300, with the Accuracy, Precision and Recall being quite similar to Trial 4, which was the easiest configuration. This could be explained in the regions with more nerve fibres,

where it is easier for the participant to recognise the target stimuli with the same spacing configuration, according to the homunculus (Kabat-Zinn, 2018).

Recommend setup locations for optimal extraction of P300 ERP:

From the results, in order to elicit P300 using tactors on one arm, spreading the tactors evenly as far from each other as possible is recommended. If both arms are involved, some adjustments to make the oddball paradigm more challenging should be considered, such as decreasing the time between stimuli and decreasing the vibrating period of each stimulus;... With the findings from Trial 5, placing the tactors in regions with dense nerve fibre could be a viable option. Some interesting configurations could be under the palm of one hand or both hands, under each finger. These configurations could provide more design approaches for the BCI wheelchair, such as integrated tactors on the armrest.

Possible limitations and suggestions

With a limited timeframe, the project had some limitations that might strengthen the findings and minimise errors in the experiment. The first limitation was that multiple trials were conducted on only one participant. Therefore, to enhance the findings in this project, evidence derived from the same experiment pipeline in multiple participants should be considered.

The second limitation was that the project did not provide a normalisation method to justify the configuration of tactor locations. In this project, tactors were first placed very close to each other in the middle of the participant's arm, then slowly increased until tactors were spread all over the arm. Ideally, to generalise the recommended tactor location setup on multiple participants, the spacing between tactors and between the first and the last tactor with respect to the participant's arm length should be in percentage.

Another limitation was that the project did not evaluate the participants' mental state at the beginning and end of each trial. Since the P300 ERP experiment requires the participant to get involved in the task, factors such as stress, tiredness, lack of sleep, and hunger could affect the participant's alertness. Therefore, a set of questionnaires to evaluate the mental state before and after the experiment could provide more interesting insights into how it can affect the performance outcome of the participant.

CONCLUSIONS

The project investigated the effect of tactor localisation on how well the participant in recognising the vibration of the target tactor. While there are various studies examining the vibrotactile in P300 BCI, there is a lack of studies exploring the correlation between the tactor's placement and the performance in eliciting P300. The project was able to provide evidence of this correlation by analysing the EEG signal in 5 different configurations. It can be concluded from the findings that, using the oddball paradigm with 4 tactors, the P300 wave could be obtained in midline channels Cz, CPz, Pz, POz and Oz, with Pz channel being the most suitable location for both P300 eliciting and convenience in BCI setup. Moreover, the findings also showed that as the spacing between tactors increased, the amplitude of target responses increased, which led to larger P300 waves in the 300 - 500 ms range. Hence, the participant was able to tell the correct vibrating tactor better when they were further apart, which was also confirmed by the Confusion Matrix derived from the EMG data. From the results obtained in this project, recommendations for tactor locations set up on the participant’s arm for BCI applications were derived. If the tactors are placed in one arm, they should be spread evenly across the arm, as far apart as possible. If the tactors are placed under the palms of the hand or fingers, they could be placed relatively close to each other, and the P300 wave could be obtained. However, further investigation into tactor configurations in regions with dense nerve fibres, such as hand, and their effect on the performance of P300 eliciting should be carried out to provide more evidence.

The project also has some limitations, which could be extended as future works. The project only carried out multiple trials on 1 participant, as well as there was a lack of normalisation method to justify the tactor locations. Therefore, further studies expanding the trials on multiple participants could provide a better understanding of the tactor locations and P300 extracting performance, while introducing a normalisation method to justify the tactor locations could help achieve consistency and comparable results between participants.

FUTURE WORK

Based on the findings and limitations derived from the project, there are multiple approaches for future research to expand into more interesting investigations. One potential future research project is to investigate more variations or configurations of spacing between tactors, which could lead to a vibrotactile map for BCI applications. Another interesting study would be the effect of distractions on the experiment. The distractions could be watching videos while performing the required tasks or simulating real-world use cases of a BCI wheelchair, such as talking to a friend walking next to them or driving an electric wheelchair around. As mentioned in the previous chapters, looking into a normalisation method to justify the configuration of the tactor location could also be a good option, as it would provide a more reliable method to compare the results between participants to find a meaningful relationship. In a BCI system, a more straightforward system could help increase the accuracy, reliability and performance. Therefore, an investigation into the use of 3 tactor configurations instead of 4 tactor configurations in the performance of eliciting P300 ERP would be another interesting finding. Last but not least, an expansion of this project by investigating multiple participants can increase the population and provide more robust and more meaningful evidence that supports the findings of this project.

REFERENCES

BELHAOUARI, A., BELKACEM, A. N. & BERRACHED, N. 2018. A Tactile P300 Brain-Computer Interface: Principle and Paradigm. EasyChair.

CHATTERJEE, A., AGGARWAL, V., RAMOS, A., ACHARYA, S. & THAKOR, N. V. 2007. A brain-computer interface with vibrotactile biofeedback for haptic information. *Journal of neuroengineering and rehabilitation,* 4**,** 1-12.

DUAN, X., GUO, S., CHEN, L. & WANG, M. 2023. A P300 Brain-Computer Interface for Lower Limb Robot Control Based on Tactile Stimulation. *Journal of Medical and Biological Engineering,* 43**,** 22-31.

FLEURY, M., LIOI, G., BARILLOT, C. & LéCUYER, A. 2020. A survey on the use of haptic feedback for brain-computer interfaces and neurofeedback. *Frontiers in Neuroscience,* 14**,** 528.

HAJIAN, A. & YONG, S.-P. Feature extraction from EEG data for a P300 based brain-computer interface. Trends and Applications in Knowledge Discovery and Data Mining: PAKDD 2017 Workshops, MLSDA, BDM, DM-BPM Jeju, South Korea, May 23, 2017, Revised Selected Papers 21, 2017. Springer, 39-50.

HOFFMANN, U., VESIN, J.-M., EBRAHIMI, T. & DISERENS, K. 2008. An efficient P300-based brain–computer interface for disabled subjects. *Journal of Neuroscience methods,* 167**,** 115-125.

HUANG, X., LIANG, S., LI, Z., LAI, C. Y. Y. & CHOI, K.-S. 2022. EEG-based vibrotactile evoked brain-computer interfaces system: a systematic review. *Plos one,* 17**,** e0269001.

IM, C.-H. 2018. Computational EEG analysis. *Springer Singap Singap,* 10**,** 978-981.

KABAT-ZINN, J. 2018. Homunculus. *Mindfulness,* 9**,** 1974-1978.

KAISER, D. A. 2005. Basic principles of quantitative EEG. *Journal of Adult Development,* 12**,** 99-104.

KIM, K.-T., SUK, H.-I. & LEE, S.-W. 2016. Commanding a brain-controlled wheelchair using steady-state somatosensory evoked potentials. *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* 26**,** 654-665.

KONO, S. & RUTKOWSKI, T. M. 2015. Tactile-force brain-computer interface paradigm: Somatosensory multimedia neurotechnology application. *Multimedia Tools and Applications,* 74**,** 8655-8667.

MAK, J., ARBEL, Y., MINETT, J. W., MCCANE, L. M., YUKSEL, B., RYAN, D., THOMPSON, D., BIANCHI, L. & ERDOGMUS, D. 2011. Optimizing the P300-based brain–computer interface: current status, limitations and future directions. *Journal of neural engineering,* 8**,** 025003.

MIN, B.-K., MARZELLI, M. J. & YOO, S.-S. 2010. Neuroimaging-based approaches in the brain–computer interface. *Trends in biotechnology,* 28**,** 552-560.

MORI, H., MATSUMOTO, Y., KRYSSANOV, V., COOPER, E., OGAWA, H., MAKINO, S., STRUZIK, Z. R. & RUTKOWSKI, T. M. Multi-command tactile brain computer interface: A feasibility study. Haptic and Audio Interaction Design: 8th International Workshop, HAID 2013, Daejeon, Korea, April 18-19, 2013, Revised Selected Papers 8, 2013. Springer, 50-59.

NEGI, T. Analysis and processing of EEG signal: A review. Proceedings of the International Conference on Smart Data Intelligence (ICSMDI 2021), 2021.

NICOLAS-ALONSO, L. F. & GOMEZ-GIL, J. 2012. Brain computer interfaces, a review. *sensors,* 12**,** 1211-1279.

ORTNER, R., DINARèS-FERRAN, J., IRIMIA, D.-C. & GUGER, C. Towards Improved Vibro-Tactile P300 BCIs. Human-Computer Interaction. Interaction Techniques and Novel Applications: Thematic Area, HCI 2021, Held as Part of the 23rd HCI International Conference, HCII 2021, Virtual Event, July 24–29, 2021, Proceedings, Part II 23, 2021. Springer, 65-74.

PéREZ-VIDAL, A. F., GARCIA-BELTRAN, C. D., MARTíNEZ-SIBAJA, A. & POSADA-GóMEZ, R. 2018. Use of the stockwell transform in the detection of P300 evoked potentials with low-cost brain sensors. *Sensors,* 18**,** 1483.

PICTON, T. W. 1992. The P300 wave of the human event-related potential. *Journal of clinical neurophysiology,* 9**,** 456-456.
RAMADAN, R. A. & VASILAKOS, A. V. 2017. Brain computer interface: control signals review. *Neurocomputing,* 223**,** 26-44.
SAHA, S., MAMUN, K. A., AHMED, K., MOSTAFA, R., NAIK, G. R., DARVISHI, S., KHANDOKER, A. H. & BAUMERT, M. 2021. Progress in brain computer interface: Challenges and opportunities. *Frontiers in Systems Neuroscience,* 15**,** 578875.
SUBHA, D. P., JOSEPH, P. K., ACHARYA U, R. & LIM, C. M. 2010. EEG signal analysis: a survey. *Journal of medical systems,* 34**,** 195-212.
SUR, S. & SINHA, V. K. 2009. Event-related potential: An overview. *Industrial psychiatry journal,* 18**,** 70.
VAID, S., SINGH, P. & KAUR, C. EEG signal analysis for BCI interface: A review. 2015 fifth international conference on advanced computing & communication technologies, 2015. IEEE, 143-147.
ZABCIKOVA, M., KOUDELKOVA, Z., JASEK, R. & LORENZO NAVARRO, J. J. 2022. Recent advances and current trends in brain-computer interface research and their applications. *International Journal of Developmental Neuroscience,* 82**,** 107-123.
ZHANG, G., GARRETT, D. R. & LUCK, S. J. 2024. Optimal filters for ERP research II: Recommended settings for seven common ERP components. *Psychophysiology***,** e14530.

APPENDICES

A. Target and Non-Target responses in midline channels

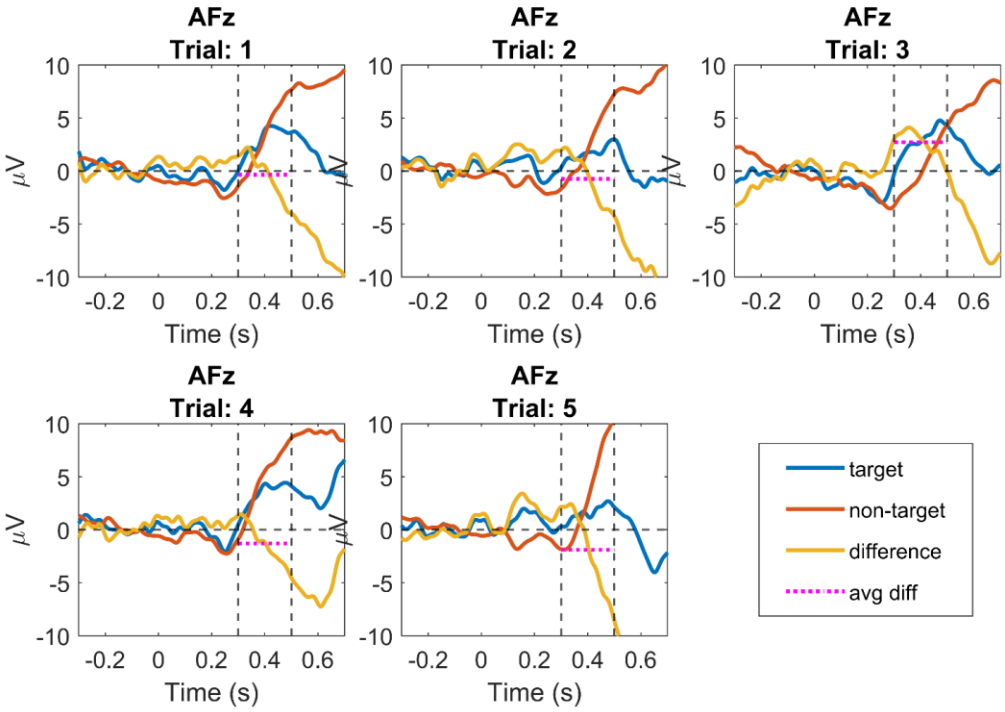

Figure 8A – Target, Non-Target responses, and their difference line in each trial of the AFz channel.

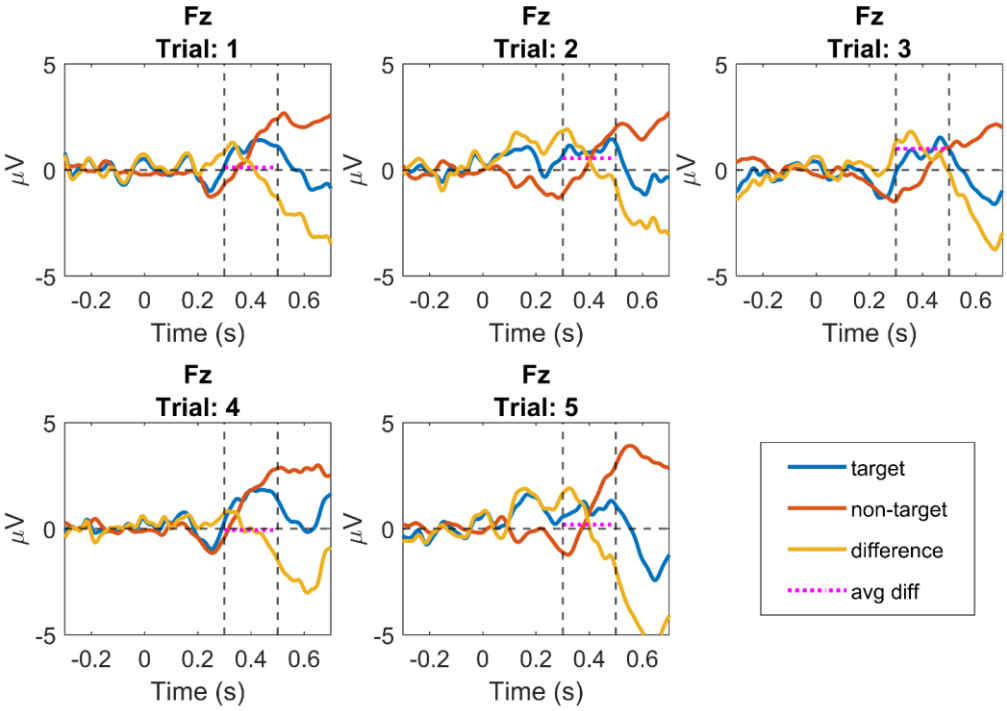

Figure 9A – Target, Non-Target responses, their difference line in each trial of the Fz channel.

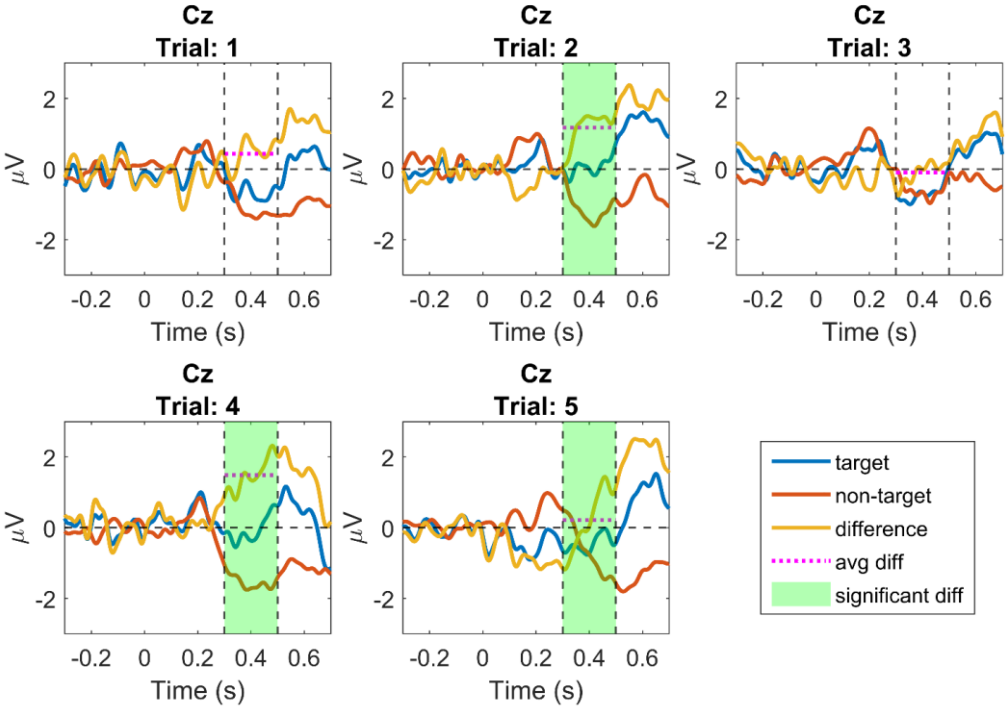

Figure 10A – Target and Non-Target responses, their difference line, and green patch show significant differences between Target and Non-Target responses in a t-test in each trial of the Cz channel.

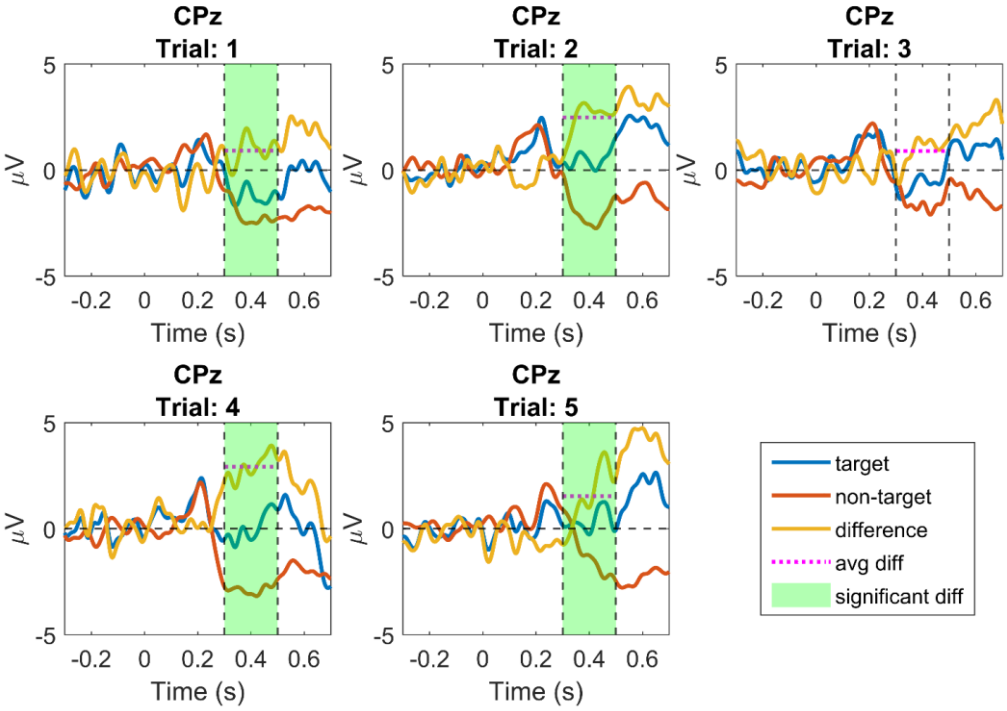

Figure 11A – Target and Non-Target responses, their difference line, and green patch show significant differences between Target and Non-Target responses in a t-test in each trial of the CPz channel.

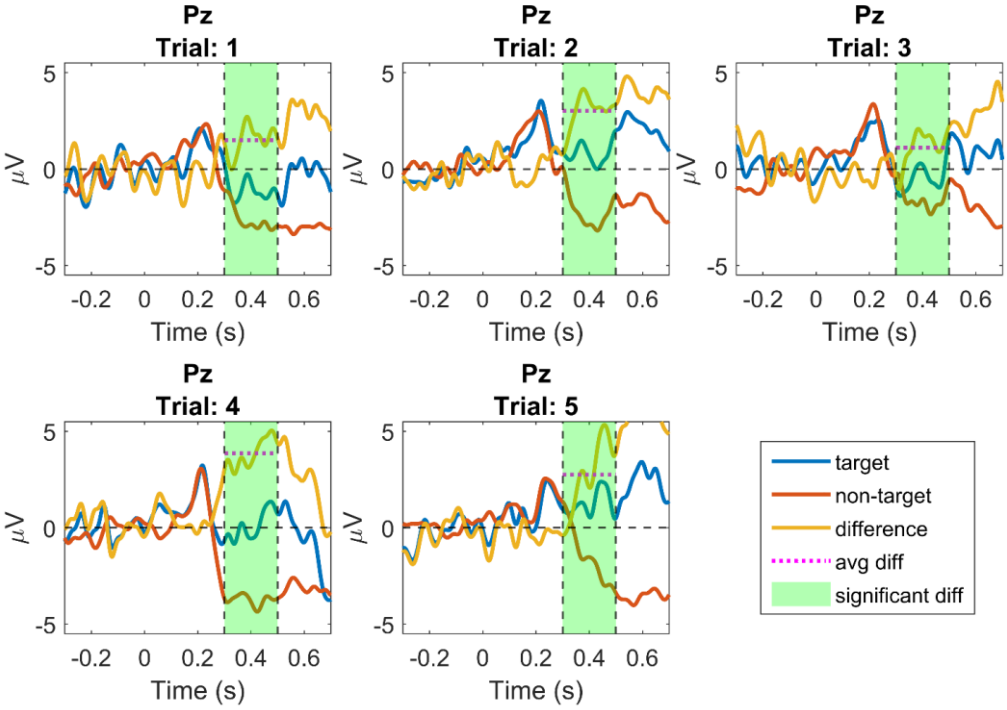

Figure 12A – Target and Non-Target responses, their difference line, and green patch show significant differences between Target and Non-Target responses in a t-test in each trial of the Pz channel.

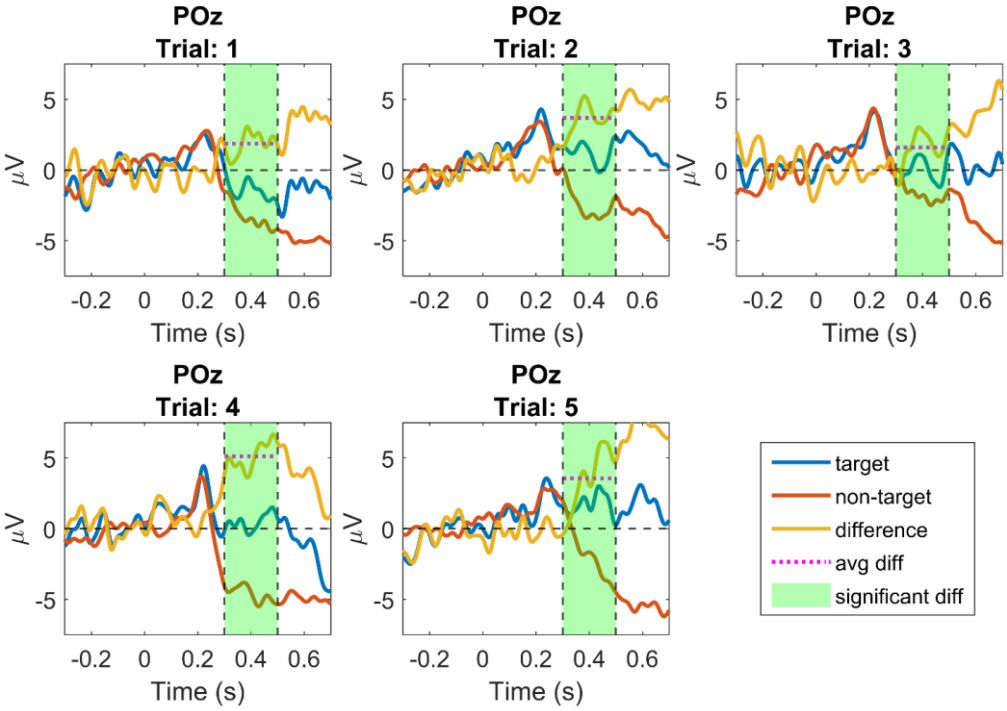

Figure 13A – Target and Non-Target responses, their difference line, and green patch show significant differences between Target and Non-Target responses in a t-test in each trial of the POz channel.

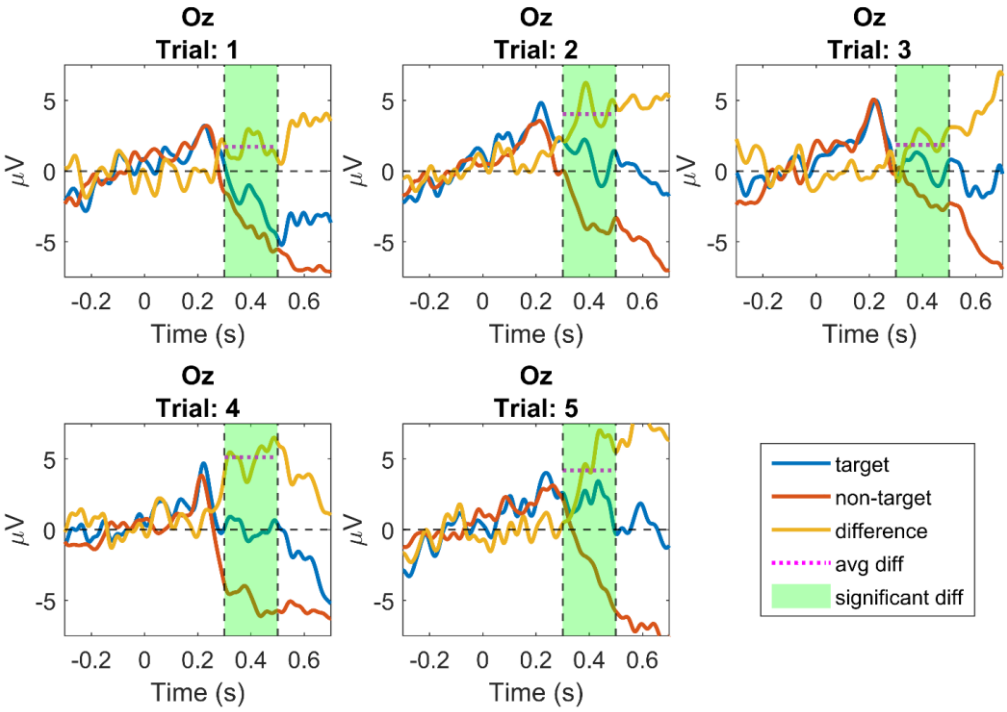

Figure 14A – Target and Non-Target responses, their difference line, and green patch show significant differences between Target and Non-Target responses in a t-test in each trial of the Oz channel.

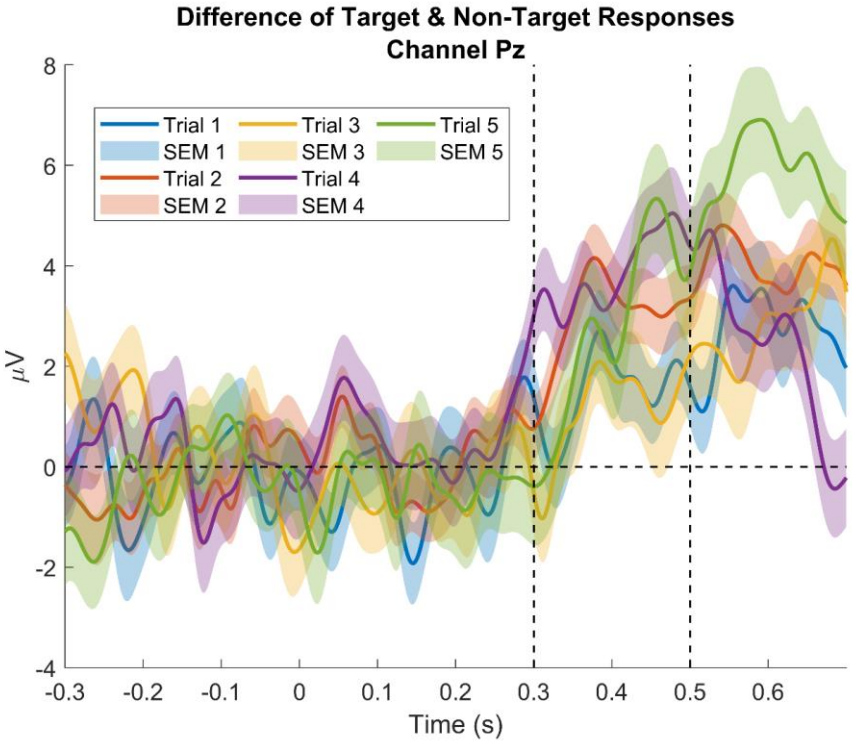

Figure 15A – Difference lines of Target and Non-Target responses and their SEM patch in all trials of the Pz channel.

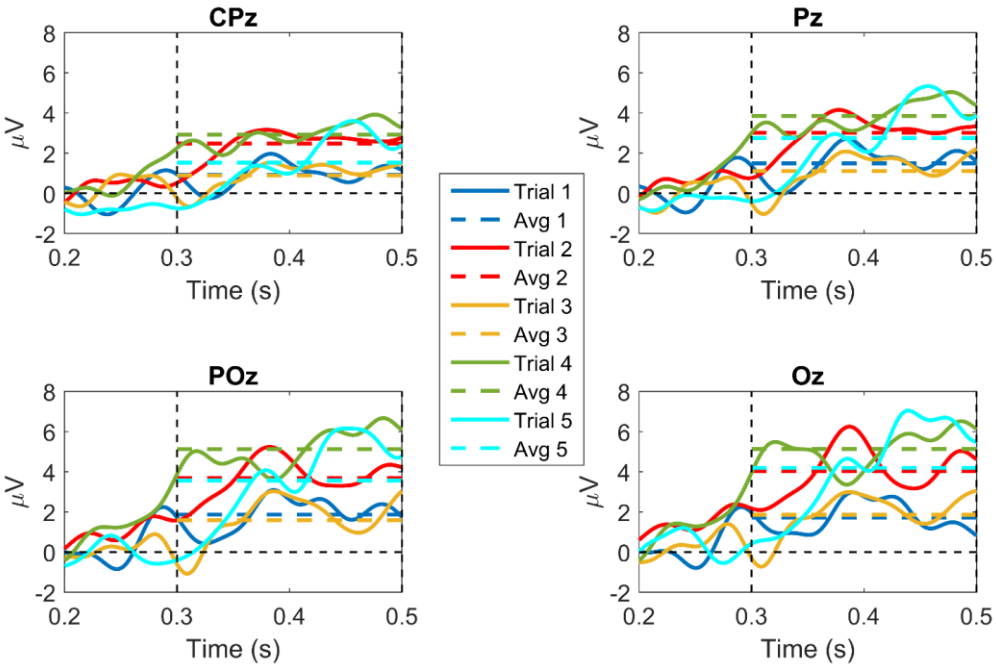

Figure 16A – Difference lines of Target and Non-Target responses and their SEM patch in all trials of the CPz, Pz, POz, Oz channels.

B. Python code for oddball paradigm using 4 tactors

```
#!/usr/bin/env python3
import keyboard
import logging
import time
import random
import numpy as np
import RPi.GPIO as GPIO

## Initial RPi configurations
logging.basicConfig(format='%(asctime)s %(levelname)-8s
%(message)s',level=logging.INFO, datefmt='%Y-%m-%d %H:%M:%S', filename =
"logv11.log", filemode = "w")
debug = False

# Define motor
motor1 = {
    "motor" : 1,
    "pin" : 22,
    "signal" : 17
}

motor2 = {
    "motor" : 2,
    "pin" : 24,
    "signal" : 27
}

motor3 = {
    "motor" : 3,
    "pin" : 23,
    "signal" : 18
}

motor4 = {
    "motor" : 4,
    "pin" : 25,
    "signal" : 4
}

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

# Set relevant pins as output
GPIO.setup(motor1["pin"], GPIO.OUT)
GPIO.setup(motor1["signal"], GPIO.OUT)
GPIO.setup(motor2["pin"], GPIO.OUT)
GPIO.setup(motor2["signal"], GPIO.OUT)
GPIO.setup(motor3["pin"], GPIO.OUT)
GPIO.setup(motor3["signal"], GPIO.OUT)
```

```
GPIO.setup(motor4["pin"], GPIO.OUT)
GPIO.setup(motor4["signal"], GPIO.OUT)

# Ensure outputs to gHiAmp are off
GPIO.output(motor1["signal"], False)
GPIO.output(motor2["signal"], False)
GPIO.output(motor3["signal"], False)
GPIO.output(motor4["signal"], False)

# Turn off motors
GPIO.output(motor1["pin"], False)
GPIO.output(motor2["pin"], False)
GPIO.output(motor3["pin"], False)
GPIO.output(motor4["pin"], False)

# Define trial parameters
W = 2
SD = 0.25
totalStim = 40 # must (mod 4 = 0)
percentTarget = 0.25
numTargetStim = int(percentTarget * totalStim)
numNonTargetStim = int((totalStim - numTargetStim)/3) # per motor
numInitStim = 5
numLastStim = numTargetStim
numBlockSets = 2 # Each block set has 4 stimulation blocks corresponding to 4
target tactors

def trialFunction():
    print("Thank you for participating in this experiment")
    print("Press Enter to continue")
    keyboard.wait('enter')
    print("You should have 4 tactors – devices that vibrate – attached to your
body")
    print("Your task is to attend to one particular tactor, and tap your index
finger when it vibrates")
    print("")
    time.sleep(W)
    print("Press Enter to continue")
    keyboard.wait('enter')
    print("")
    print("First, we will do some practice")
    print("")
    time.sleep(W)

    while True:
        introduceFunction()
        print("")
        print("Press Tab to redo the tactors introduction")
        print("Press Enter to continue to the experiment practice")
        print("")
        if keyboard.read_key() == "tab":
```

```
            continue
        elif keyboard.read_key() == "enter":
            break

    print("Now we will practice the experiment")
    print("")
    time.sleep(W)

    while True:
        practiceFunction()
        print("")
        print("Press Tab to redo the experiment practice")
        print("Press Enter to continue to the actual experiment")
        print("")
        if keyboard.read_key() == "tab":
            continue
        elif keyboard.read_key() == "enter":
            break

    print("Now we will run the experiment")
    print("")
    time.sleep(W)
    print("It will consist of", numBlockSets * 4, "blocks, with rests in between
blocks")
    print("Each block will consist of 4 tactor vibrations")
    print("It will take about 30 minutes, plus rest time")
    print("")
    time.sleep(W)
    print("Press Enter to continue")
    keyboard.wait('enter')
    print("Please sit still and relax")
    print("The experiment will start shortly")
    print("")
    time.sleep(W)
    logging.info("Start experiment")
    paradigmFunction()
    print("That completed block", numBlockSets * 4, "and the experiment is
completed")
    logging.info("End experiment")
    print("Please relax and wait for the technician to attend to you")
    print("Thank you for your participation")

def introduceFunction():
    for i in range(1):
        print("Each tactor will vibrate, one after the other")
        time.sleep(W)
        print("Press Enter to start")
        keyboard.wait('enter')
        print("")
        print("Tactors are about to vibrate")
        time.sleep(W)
```

```
        print("")
        logging.info("Start introducing")
        print("Tactor 1 is vibrating")
        onOff(motor1)
        time.sleep(W)
        print("Tactor 2 is vibrating")
        onOff(motor2)
        time.sleep(W)
        print("Tactor 3 is vibrating")
        onOff(motor3)
        time.sleep(W)
        print("Tactor 4 is vibrating")
        onOff(motor4)
        time.sleep(W)
        logging.info("End introducing")
        logging.info("")

def practiceFunction():
    print("Press Enter to start practice block")
    print("")
    keyboard.wait('enter')
    print("Starting practice block")
    print("")
    logging.info("Starting practice block")
    time.sleep(W)
    print("The target tactor and vibration orders are random in each block")
    time.sleep(W)
    print("You need to focus on target tactor")
    time.sleep(W)
    print("Tap your index finger when you think that tactor is vibrating")
    print("")
    time.sleep(W)
    print("Press Enter to initialize the practice block")
    print("")
    keyboard.wait('enter')

    ########################################################################
    #.......................Motor Order Shuffle.......................#
    ########################################################################

    motorIndex = np.array([1, 2, 3, 4])
    random.shuffle(motorIndex)
    logging.info("Motor index: %s", motorIndex)

    for i in range(1):
        motorTarget = motorIndex[i] # Put target motor in a variable
        motorNonTarget = np.delete(motorIndex, i) # Put non-target motors in an
array

        ########################################################################
        #.......................Random Generator.......................#
```

```
        ######################################################################
        # Create the stimuli array
        arrayNonTargets =
np.array([motorNonTarget[0]]*numNonTargetStim+[motorNonTarget[1]]*numNonTargetSt
im+[motorNonTarget[2]]*numNonTargetStim)
        random.shuffle(arrayNonTargets)
        initStim = arrayNonTargets[0:numInitStim]
        middleStim = arrayNonTargets[numInitStim:(arrayNonTargets.size -
numLastStim)]
        lastStim = arrayNonTargets[(arrayNonTargets.size - numLastStim):]
        arrayTargets = np.array([motorTarget]*numTargetStim)
        arrayNoInit = np.concatenate((middleStim,arrayTargets))
        random.shuffle(arrayNoInit)
        arrayStim = np.concatenate((initStim,arrayNoInit))

        l = 0
        for m in range(totalStim):
            if (arrayStim[m] == motorTarget):
                arrayStim = np.insert(arrayStim,m+1,lastStim[l])
                m = m+1
                l += 1

        # Start the trials

        print("We will now initialize the practice block")
        print("")
        time.sleep(W)
        print("All tactors will vibrate to initialize the block")
        print("")
        time.sleep(4)
        logging.info("Initialize the practice block, tactors vibrate")
        onOff(motor1)
        onOff(motor4)
        onOff(motor2)
        onOff(motor3)
        logging.info("Marker practice")
        time.sleep(W)
        print("You need to focus on tactor", [motorTarget], "(which is
vibrating)")
        time.sleep(W)

        for p in range(0,3): # vibrate target tactor 3 times
            if motorTarget == 1:
                onOff(motor1)
            elif motorTarget == 2:
                onOff(motor2)
            elif motorTarget == 3:
                onOff(motor3)
            else:
                onOff(motor4)
            time.sleep(0.5)
```

```
        print("Tap your index finger when you think tactor", [motorTarget], "is
vibrating")
        time.sleep(W)
        print("Keep focusing at the screen during the block")
        print("")
        time.sleep(W)
        print("Press Tab to vibrate the target tactor that you need to focus")
        print("Press Enter when you are ready to start the practice block")
        print("")

        while True:
            if keyboard.read_key() == "tab":
                if motorTarget == 1:
                    onOff(motor1)
                elif motorTarget == 2:
                    onOff(motor2)
                elif motorTarget == 3:
                    onOff(motor3)
                else:
                    onOff(motor4)
            elif keyboard.read_key() == "enter":
                break

        print("Place you hand on the table and relax")
        print("")
        time.sleep(W)
        print("The block will start now")
        print("")
        logging.info("Motor target: %s", motorTarget)
        logging.info("Motor non-target: %s", motorNonTarget)
        logging.info("Stimuli array: %s", arrayStim)
        logging.info("")

        ##########################################################################
####
        #.........................Play
Stimulation.........................#
        ##########################################################################
####

        time.sleep(2)
        countTarget = 0
        countNonTarget = 0

        for m in range(totalStim):
            currentStim = arrayStim[m]
            ISI = random.uniform(0.9, 1.5)

            if currentStim == motorTarget:
                if motorTarget == 1:
```

```
                    onOff(motor1)
                elif motorTarget == 2:
                    onOff(motor2)
                elif motorTarget == 3:
                    onOff(motor3)
                else:
                    onOff(motor4)

                logging.info("Target: %s", currentStim)
                logging.info("ISI: %s", ISI)
                countTarget += 1
                time.sleep(ISI)

            else:
                if currentStim == 1:
                    onOff(motor1)
                elif currentStim == 2:
                    onOff(motor2)
                elif currentStim == 3:
                    onOff(motor3)
                else:
                    onOff(motor4)

                logging.info("Non-Target: %s", currentStim)
                logging.info("ISI: %s", ISI)
                countNonTarget += 1
                time.sleep(ISI)

    print("That completes practice block")
    print("All tactors will vibrate to finalize the block")
    print("")
    time.sleep(W)
    logging.info("")
    logging.info("Finished practice block, tactors vibrate")
    logging.info("Targets count: %s", countTarget)
    logging.info("Non-Targets count: %s", countNonTarget)
    logging.info("")
    onOff(motor1)
    onOff(motor4)
    onOff(motor3)
    onOff(motor2)
    logging.info("Marker practice")
    time.sleep(W)

def paradigmFunction():
    print("The target tactor and vibration orders are random in each block")
    time.sleep(W)
    print("You need to focus on target tactor")
    time.sleep(W)
    print("Tap your index finger when you think that tactor is vibrating")
    print("")
```

```
    print("Press Enter to start the experiment")
    print("")
    keyboard.wait('enter')

    ###########################################################################
    #.......................Motor Order Shuffle.......................#
    ###########################################################################

    motorIndex = np.array([],dtype=int)

    for i in range(numBlockSets):
        motor = np.array([1, 2, 3, 4])
        random.shuffle(motor)
        motorIndex = np.concatenate((motorIndex,motor))

    logging.info("Motor index: %s", motorIndex)
    logging.info("")

    blockIndex = 0

    for j in range(len(motorIndex)):
        blockIndex += 1
        motorTarget = motorIndex[j]
        motorNonTarget = np.array([],dtype=int)

        for k in range(len(motor)):
            if motorIndex[k] != motorTarget:
                motorNonTarget = np.append(motorNonTarget, motorIndex[k])

        ###########################################################################
        #.......................Random Generator.......................#
        ###########################################################################

        arrayNonTargets =
np.array([motorNonTarget[0]]*numNonTargetStim+[motorNonTarget[1]]*numNonTargetSt
im+[motorNonTarget[2]]*numNonTargetStim)
        random.shuffle(arrayNonTargets)
        initStim = arrayNonTargets[0:numInitStim]
        middleStim = arrayNonTargets[numInitStim:(arrayNonTargets.size -
numLastStim)]
        lastStim = arrayNonTargets[(arrayNonTargets.size - numLastStim):]
        arrayTargets = np.array([motorTarget]*numTargetStim)
        arrayNoInit = np.concatenate((middleStim,arrayTargets))
        random.shuffle(arrayNoInit)
        arrayStim = np.concatenate((initStim,arrayNoInit))

        l = 0
        for m in range(totalStim):
            if (arrayStim[m] == motorTarget):
                arrayStim = np.insert(arrayStim,m+1,lastStim[l])
                m = m+1
```

```
        l += 1

print("")
print("We will now start block", blockIndex)
time.sleep(W)
print("All tactors will vibrate to initialize the block")
print("")
time.sleep(W)
logging.info("Initialize the block, tactors vibrate")

if j == 0:
    onOff(motor1)
    onOff(motor2)
    onOff(motor3)
    onOff(motor4)
    logging.info("Marker block %s", blockIndex)
elif j == 1:
    onOff(motor2)
    onOff(motor1)
    onOff(motor3)
    onOff(motor4)
    logging.info("Marker block %s", blockIndex)
elif j == 2:
    onOff(motor1)
    onOff(motor2)
    onOff(motor4)
    onOff(motor3)
    logging.info("Marker block %s", blockIndex)
elif j == 3:
    onOff(motor2)
    onOff(motor1)
    onOff(motor4)
    onOff(motor3)
    logging.info("Marker block %s", blockIndex)
elif j == 4:
    onOff(motor4)
    onOff(motor3)
    onOff(motor2)
    onOff(motor1)
    logging.info("Marker block %s", blockIndex)
elif j == 5:
    onOff(motor3)
    onOff(motor4)
    onOff(motor2)
    onOff(motor1)
    logging.info("Marker block %s", blockIndex)
elif j == 6:
    onOff(motor4)
    onOff(motor3)
    onOff(motor1)
    onOff(motor2)
```

```
            logging.info("Marker block %s", blockIndex)
        elif j == 7:
            onOff(motor3)
            onOff(motor4)
            onOff(motor1)
            onOff(motor2)
            logging.info("Marker block %s", blockIndex)

        time.sleep(W)
        print("You need to focus on tactor", [motorTarget], "(which is
vibrating)")
        time.sleep(W)

        for p in range(0,3): # vibrate target tactor 3 times
            if motorTarget == 1:
                onOff(motor1)
            elif motorTarget == 2:
                onOff(motor2)
            elif motorTarget == 3:
                onOff(motor3)
            else:
                onOff(motor4)
            time.sleep(0.5)

        print("Tap your index finger when you think tactor", [motorTarget], "is
vibrating")
        time.sleep(W)
        print("Keep focusing at the screen during the block")
        print("")
        time.sleep(W)
        print("Press Tab to vibrate the target tactor that you need to focus")
        print("Press Enter when you are ready to start the block", [blockIndex])
        print("")

        while True:
            if keyboard.read_key() == "tab":
                if motorTarget == 1:
                    onOff(motor1)
                elif motorTarget == 2:
                    onOff(motor2)
                elif motorTarget == 3:
                    onOff(motor3)
                else:
                    onOff(motor4)
            elif keyboard.read_key() == "enter":
                break

        print("The block will start now")
        print("")
        logging.info("Start block %s", blockIndex)
        logging.info("Target motor: %s", motorTarget)
```

```
logging.info("Non-Targert motors: %s", motorNonTarget)
logging.info("Stimuli array %s", arrayStim)
logging.info("")

#########################################################################
#.........................Stimulation.............................#
#########################################################################

time.sleep(2)
countTarget = 0
countNonTarget = 0

for n in range(totalStim):
    currentStim = arrayStim[n]
    ISI = random.uniform(0.9, 1.5)

    if currentStim == motorTarget:
        if motorTarget == 1:
            onOff(motor1)
        elif motorTarget == 2:
            onOff(motor2)
        elif motorTarget == 3:
            onOff(motor3)
        else:
            onOff(motor4)

        logging.info("Target: %s", currentStim)
        logging.info("ISI: %s", ISI)
        countTarget += 1
        time.sleep(ISI)

    else:
        if currentStim == 1:
            onOff(motor1)
        elif currentStim == 2:
            onOff(motor2)
        elif currentStim == 3:
            onOff(motor3)
        else:
            onOff(motor4)

        logging.info("Non-Target: %s", currentStim)
        logging.info("ISI: %s", ISI)
        countNonTarget += 1
        time.sleep(ISI)

print("That completes block",blockIndex)
print("All tactors will vibrate to finalize the block")
print("")
time.sleep(W)
logging.info("")
```

```
logging.info("End block %s", blockIndex)
logging.info("Targets count: %s", countTarget)
logging.info("Non-Targets count: %s", countNonTarget)

if j == 0:
    onOff(motor1)
    onOff(motor3)
    onOff(motor2)
    onOff(motor4)
    logging.info("Marker block %s", blockIndex)
elif j == 1:
    onOff(motor2)
    onOff(motor3)
    onOff(motor1)
    onOff(motor4)
    logging.info("Marker block %s", blockIndex)
elif j == 2:
    onOff(motor1)
    onOff(motor3)
    onOff(motor4)
    onOff(motor2)
    logging.info("Marker block %s", blockIndex)
elif j == 3:
    onOff(motor2)
    onOff(motor3)
    onOff(motor4)
    onOff(motor1)
    logging.info("Marker block %s", blockIndex)
elif j == 4:
    onOff(motor4)
    onOff(motor2)
    onOff(motor3)
    onOff(motor1)
    logging.info("Marker block %s", blockIndex)
elif j == 5:
    onOff(motor3)
    onOff(motor2)
    onOff(motor4)
    onOff(motor1)
    logging.info("Marker block %s", blockIndex)
elif j == 6:
    onOff(motor4)
    onOff(motor2)
    onOff(motor1)
    onOff(motor3)
    logging.info("Marker block %s", blockIndex)
elif j == 7:
    onOff(motor3)
    onOff(motor2)
    onOff(motor1)
    onOff(motor4)
```

```
            logging.info("Marker block %s", blockIndex)

        logging.info("")
        time.sleep(W)
        print("Please relax and have a rest")
        print("")
        time.sleep(W)
        print("When you are ready, Press Enter to continue to next block")
        keyboard.wait('enter')
    return

##############################################################################
#########################
#Motor control function
##############################################################################
#########################
def onOff(motor):
    if (debug):
        print("[DEBUG] Activating motor:", motor["motor"], "on pin:",
motor["pin"]) # For debug purpose
    else:
        GPIO.output(motor["pin"], True)
        GPIO.output(motor["signal"], True)
        time.sleep(SD)
        GPIO.output(motor["signal"], False)
        GPIO.output(motor["pin"], False)
    return
```

C. Matlab code for data analysis

```
%% script for exploring the vibrotactile data
%% cleaning
clc;
clear;

%% definitions
%folder = '/Volumes/CSE-EEG/People/Anh/Experiment/Final/'; % Linux
folder = 'R:\CSE-EEG\People\Anh\Experiment\Final';
data_file_1 = 'trial_1.xdf';
data_file_2 = 'trial_2.xdf';
data_file_3 = 'trial_3.xdf';
data_file_4 = 'trial_4.xdf';
data_file_5 = 'trial_5.xdf';
all_stimuli_codes = { '1', '2', '4', '8'};
time_offsets = [ -0.3 0.7];
P300start = 0.3;
P300stop = 0.5;
alpha = 0.05;
lowpass_corner = 25;
highpass_corner = 0.1;
patch_args = { 'FaceAlpha', 0.3, 'EdgeColor', 'none'};

% channel
```

```
display_channels = { 'AFz', 'Fz', 'FC1', 'FC2', 'Cz', 'C1', 'C3', 'C2', 'C4', 'CPz',
'CP1', 'CP2', 'Pz', 'P3', 'P7', 'P4', 'P8', 'POz', 'Oz', 'O1', 'O2'};

EMG_channel = 'EMG';

% event code information
block_start_codes = [124 118 123 117 101 107 102 108];
block_stop_codes = [122 116 121 115 103 109 104 110];
practice_start_code = 120;
practice_stop_code = 119;
baseline_start = -0.2;
baseline_stop = 0;

% derived
Nblocks = numel( block_start_codes);


%% load the data, recode the events, and filter

df1 = [];
df2 = [];
% load data
d = eeg3.eeg.load( fullfile( folder, {data_file_1, data_file_2, data_file_3,
data_file_4, data_file_5}));

for di = 1:numel( d)
    d( di) = vibrotactile_recode_events( d( di));

    % filter data
    df1 = [ df1 d( di).highpass( highpass_corner)];
    df2 = [ df2 df1( di).lowpass( lowpass_corner)];

    % check the event code after recoding
    [d( di).event.code]
end

%% loop over each block

% make space
%{ 'AFz', 'Fz', 'FC1', 'FC2', 'Cz', 'C1', 'C3', 'C2', 'C4', 'CPz', 'CP1', 'CP2',
'Pz', 'P3', 'P7', 'P4', 'P8', 'POz', 'Oz', 'O1', 'O2'}
chan_plot = 13; % 1 2 5 10 13 18 19
figure( ), clf;
color = { '#0072BD', '#FF0000', '#EDB120', '#77AC30', '#00FFFF'};
% colour_order = get( gca, 'ColorOrder');
colour_order = [ 0    0.4470    0.7410; 0.8500    0.3250    0.0980];
hold on

% loop over blocks
ha = zeros( 1, numel( display_channels));
for dfi = 1:numel( df2)
    eft = [];
    efnt = [];

    for blocki = 1:Nblocks
    % extract the data for the current block
        b = df2( dfi).eventslice( block_start_codes( blocki), block_stop_codes(
blocki), ...
        d( dfi).time.step * [ -1 1]);

        % check the event code after slicing
        [b.event.code]
```

```
        % find the target tactor
        target_tactor_label = int2str( b.event( 2).code - 200);
        non_target_tactor_labels = setdiff( all_stimuli_codes, target_tactor_label);

        % remove the target stimuli that are initiated by the participant
        for i = 1:numel([b.event.code])
            if [b.event( 3).code] == str2double(target_tactor_label)
                b.event( 3) = [];
            else
                break
            end
        end

        % check the event code after removing unwanted target stimuli
        [b.event.code]

        % extract target tactor and non-target tactor epochs, and average
        eft = [ eft b.extractepochs( target_tactor_label, time_offsets( 1),
time_offsets( 2))];
        efnt = [ efnt b.extractepochs( non_target_tactor_labels, time_offsets( 1),
time_offsets( 2))];
    end

    % mef = [ eft.mean, efnt.mean, eft.mean - efnt.mean];
    % mef = [ eft.mean, efnt.mean];
    % mef = [ eft.mean - efnt.mean];
    mef = [ eft.mean];
    mef = mef.subbaseline( baseline_start, baseline_stop);

    % display results
    P300At = [ eft.selectchan( display_channels{ chan_plot}).selecttime( ...
        P300start, P300stop).meantime.data];
    P300Ant = [ efnt.selectchan( display_channels{ chan_plot}).selecttime( ...
        P300start, P300stop).meantime.data];
    [h,p] = ttest2( P300At, P300Ant)
    % mean_diff = mean(mef( 3).selectchan( display_channels{ chan_plot}).selecttime(
P300start, P300stop).data);
    % mean_diff = mean( mef.selectchan( display_channels{ chan_plot}).selecttime(
P300start, P300stop).data);
%     subplot( 2, round(numel( df2) / 2), dfi);
%     if ha( dfi) == 0
%         ha( dfi) = subplot( 2, round(numel( df2) / 2), dfi);
%     end
%     axes( ha( dfi));
%     mef.selectchan( display_channels{ chan_plot}).plot( 'Linewidth', 2);
    mef.selectchan( display_channels{ chan_plot}).plot( 'Color', color{ dfi},
'Linewidth', 2);
    % plot([ P300start; P300stop], [ mean_diff, mean_diff], ':', 'Color', color{
dfi}, 'LineWidth', 2);
    hold on
%     for linei = 1:numel( mef)
%         Y = mef( linei).selectchan( display_channels{ chan_plot}).data;
%         E = mef( linei).userdata.sem;
%         E = E( mef( linei).chan.findchan( display_channels{ chan_plot}), :);
%         patch( [ mef( linei).timevector fliplr( mef( linei).timevector)], ...
%             [ Y + E, fliplr( Y - E)], colour_order( linei, :), ...
%             patch_args{ :});
%     end
%     ylim([ -4 8]);
    yline(0, '--', 'Color', 'black', 'LineWidth', 1, 'HandleVisibility','off');
    xline(0.3, '--', 'Color', 'black', 'LineWidth', 1, 'HandleVisibility','off');
```

```
    xline(0.5, '--', 'Color', 'black', 'LineWidth', 1, 'HandleVisibility','off');
    %yticks( [mean_diff, ylims]);
    title( {['Target Responses']; ['Channel ', display_channels{ chan_plot}]})
%     title( {['Trial ', int2str(dfi)]; ['Channel ', display_channels{
chan_plot}]});
end

% force all axes to have the same scale
% for dfi = 1:numel( display_channels)
%     switch dfi
%         case 1
%             set( ha( dfi), 'Position', [ 0.08 0.5838 0.2635 0.28]);
%         case 2
%             set( ha( dfi), 'Position', [ 0.40 0.5838 0.2635 0.28]);
%         case 3
%             set( ha( dfi), 'Position', [ 0.73 0.5838 0.2635 0.28]);
%         case 4
%             set( ha( dfi), 'Position', [ 0.08 0.1100 0.2635 0.28]);
%         case 5
%             set( ha( dfi), 'Position', [ 0.40 0.1100 0.2635 0.28]);
%     end
% end
% hl = legend({ 'target', 'non-target', 'difference', 'avg diff', 'significant
diff'}, 'Location', [ 0.8 0.11 0.26 0.28]);
% hl = legend({'Difference', 'SEM'}, 'Location', [ 0.8 0.11 0.10 0.15]);
% hl = legend({'Target', 'Non-Target', 'SEM Target', 'SEM Non-Target'}, 'Location',
[ 0.8 0.11 0.10 0.15]);
% hl = legend({'Trial 1', 'SEM 1', 'Trial 2', 'SEM 2', 'Trial 3', 'SEM 3', 'Trial
4', 'SEM 4', 'Trial 5', 'SEM 5'}, 'Location', [ 0.315 0.13 0.10 0.15]); % (0.15 &
0.7)
hl = legend({'Trial 1', 'Trial 2', 'Trial 3', 'Trial 4', 'Trial 5'}, 'Location', [
0.315 0.13 0.10 0.15]); % (0.15 & 0.7)
% hl = legend({'Trial 1', 'Avg 1', 'Trial 2', 'Avg 2', 'Trial 3', 'Avg 3', 'Trial
4', 'Avg 4', 'Trial 5', 'Avg 5'}, 'Location', [ 0.315 0.7 0.10 0.15]);
%legend( { 'target', 'non-target', 'difference', 'significant diff'}, 'Location',
'SouthWest');
hl.NumColumns = 3;
linkaxes( findobj( gcf, 'Type', 'axes'));

pretty_print( 'target', 'thesis', 'filename', 'delete_me555.png')
```