Politecnico di Torino

Master's degree in COMPUTER ENGINEERING

Master's Degree Thesis

Emotion Recognition Application on Desktop Application

Supervisors

Prof. Gabriella OLMO

Prof. Vito DE FEO

Candidate

Victor SEGUIN

March 2022

Abstract

Traditional marketing research seeks to consciously comprehend the decision-making process of the consumer, whereas neuromarketing focuses at understanding customer conduct and to identify various needs of the customers. The difficulty with conventional marketing research depends primarily on what the customer feels and believes. Our worldly perspective is generally guided by our emotions that are often changing. This is the reason why the project team aims to record the participant's unconscious behaviour as well as his conscious behaviour.

In this context, a desktop application was developed and its goal is to detect a participant's emotion and attention when faced to various types of neuromarketing stimuli : a static image advertisement, a video advertisement and a website-based advertisement. It manages the pathway of all experiments while recording the participant's behaviour and his attention span in the meantime.

The use of various physiological indicators called biomarkers in these three experiments allows us to get a better understanding of the participant's emotions aswell as his attention during the different experiments. Three different types of biomarkers were particularly studied for this project : eye tracking, face emotion recognition and galvanic skin resistance.

After the experiments and the recordings, a postprocessing software has been created with the aim of obtaining an analysis tool for the eye tracking data analysis, which I was the most involved in. It was used specifically for the website experiment and it was used to compare two different versions of the same website.

Finally, the statistical analysis methods for eye tracking data and face emotion recognition are explained, with a possible example of how it is

possible to detect emotions based on the participant's arousal and valence using a multiple biomarker approach like in this specific project.

Acknowledgements

During this whole project that took place during the COVID-19 pandemic, I have gotten a lot of support from my supervisors Professor Vito De Feo and Professora Gabriella Olmo.

Besides, I would like to thank all the team members I worked with who made this work an unforgettable experience.

Table of Contents

List of Figures

Acronyms

EEG : Electroencephalography

fMRI : Functional magnetic resonance imaging

GSR : Galvanic Skin Response

HCI : Human-Computer interaction

FER : Face Emotion Recognition

ASD : Autism Spectrum Disorder

FFMPEG : Fast Forward MPEG

SVM : Support Vector Machine

Glossary

Stimulus : A stimulus can be any visual content presented to participants during an eye tracking experiment. Typically, static and dynamic stimuli, with either active or passive content are differentiated. Usually, 2D stimuli are presented to participants. In this project, the stimulus can either be images, videos or websites depending on the type of experiment performed.

Gaze point : Gaze points show what the eyes are looking at. They correspond to the coordinates of the eyes at any moment of the time.

Fixation : When we observe a series of gaze points that is very close – in time and or in space – this gaze cluster constitutes a fixation, denoting a period where the eyes are locked towards an object. Fixations are oftenly used as a visual attention metrics in eye tracking studies[1].

heatmaps : they are visualizations which show the general distribution of gaze points on an image. They are typically displayed as a color gradient overlay on the presented image or stimulus. Using a heatmap is a straightforward method to quickly visualize which elements attract more attention than others. Heatmaps can be compared across single respondents as well as groups of participants, which can be helpful in understanding how different populations might view a stimulus in alternative ways.

Area of Interest : it designates a tool used to select regions of a displayed stimulus, and to extract metrics specifically for those regions. It is not a metric by itself, but an area where other metrics are extracted and calculated.

Saccade : Saccades describe a rapid eye movement from one fixation to another. They typically last about 30 to 80 ms and are the fastest movement the human body can perform.[1]

Support vector machines : They are a set of supervised learning techniques for solving discrimination and regression problems. SVMs are a generalization of linear classifiers. SVMs can be used to solve discrimination problems, i.e. deciding which class a sample belongs to, or regression problems, i.e. predicting the numerical value of a variable.

The margin is the smallest distance between the training samples and the separating hyperplane that satisfies the separability condition

The goal of the SVM is to find the more optimized margin possible, by finding the minima

$$\underset{w,b}{\text{minimize}} \frac{1}{2} \|w\|^2 \text{ subject to } y_i \left[\langle x_i, w\rangle + b\right] \geq 1$$

with :

$x = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^T$: being the input vector

$w = (\mathbf{w}_1, \ldots, \mathbf{w}_N)^T$ being the weight vector

y being the discrimination value .In case of binary discrimination

$y \in \{-1, 1\}$

Mean shift algorithm : The mean shift is a general non-parametric mode finding/clustering procedure widely used in image processing and analysis and computer vision techniques such as image denoising, image segmentation, motion tracking, etc.

arousal : caracterizes the activation/deactivation of an emotion

valence : caracterizes the degree of pleasantness/unpleasantness in an emotion.

Circumplex Model of Affects : proposed by Russell et al. [2], it consists in a distributed two-dimensional circular space comprising the axes of arousal (activation/deactivation) and valence (pleasant/unpleasant), as we can see in Figure 0.1

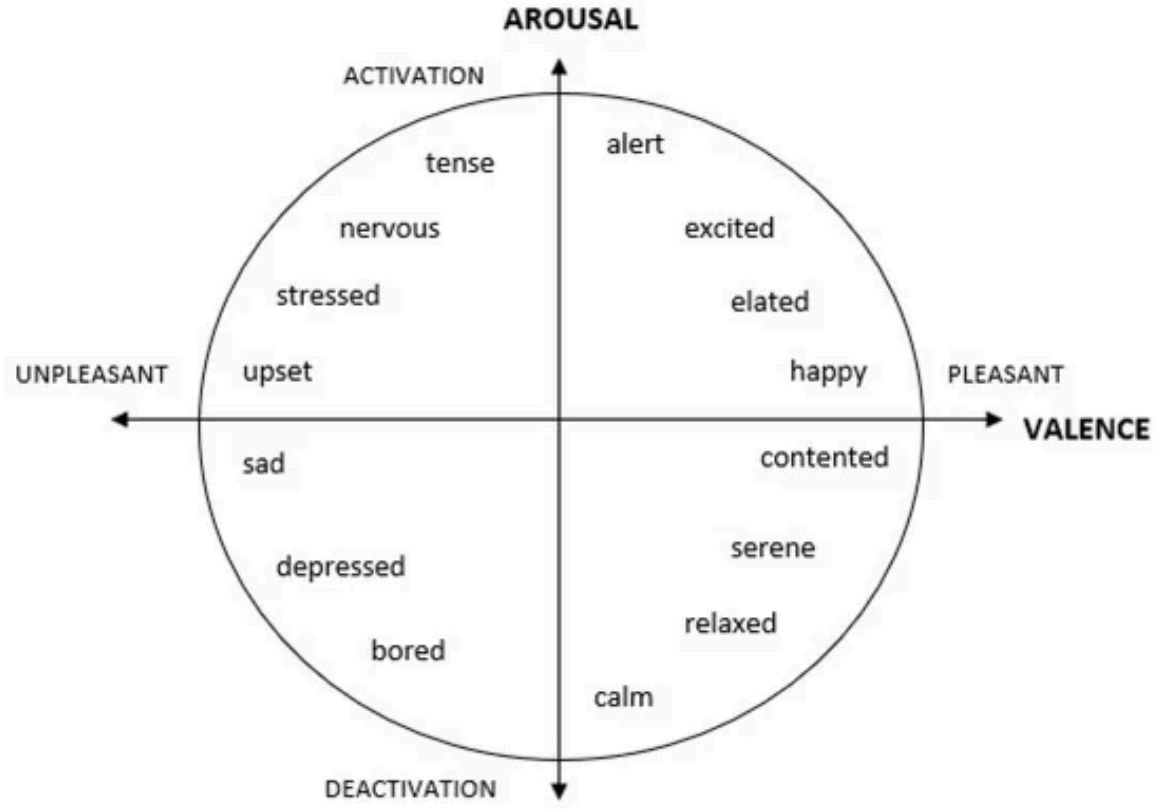


Figure 0.1 - A graphical representation of the Circumplex Model of Affect

Introduction

The aim of this master thesis is to create a desktop-based application for conducting neuroscientific experiments on part based at the Caserte hospital, the Centro Puzzle. Starting from scratch, we created an application capable of collecting data coming from various sources, in an experimental context. This kind of project can only be done by involving many people from different teams, working in different fields such as neuropsychology, marketing, computer engineering and management. I was teamed up with students from different cultures and from different schools as shown below in figure 0.1.

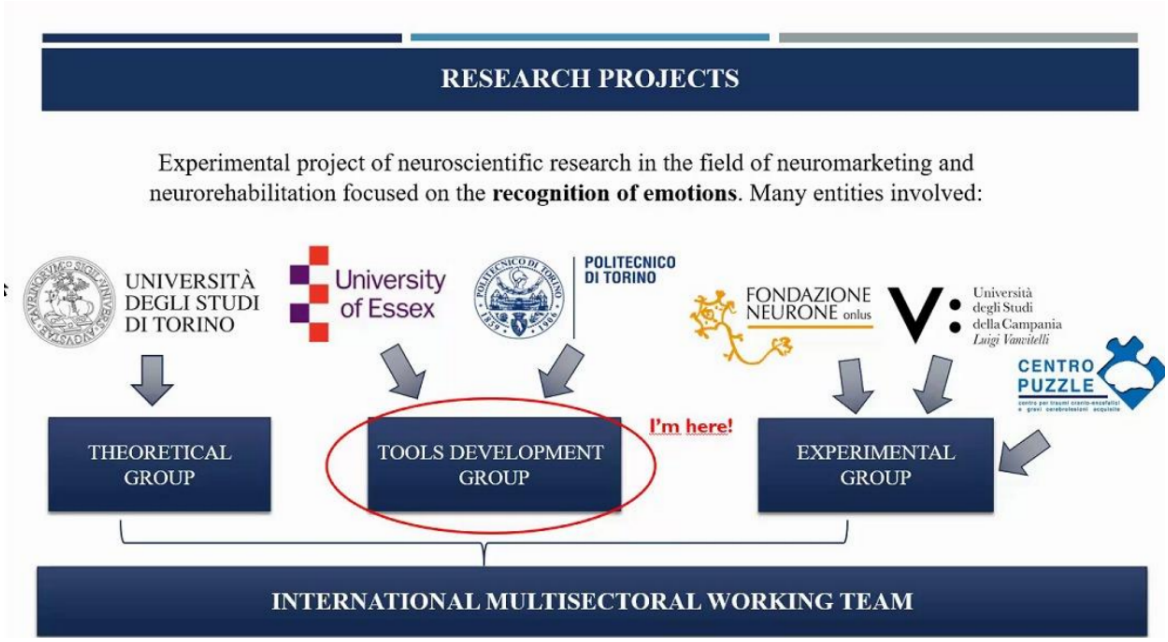


Figure 0.2 : Project overview

I worked since September 2020 under the supervision of Professor Vito DE FEO and I had the opportunity to discuss and exchange with people from different fields as mine, giving me the chance to address one specific topic which is emotion recognition from different point of views : the point of view of an engineer, the point of view of a neuropsychologist, the point of view of an operator from the hospital Centro Puzzle….

The upmost challenge in this project was to include everyone's work and expertise that is mostly theoretical onto a single application where compromises must be made, and the best implementation is not always the one that was preferred by group members from other fields. Choices are then made according to a huge set of constraints, which made the project even more challenging.

First of all, I will present in this paper an introduction to neuromarketing, the intakes of this project and the different experiments that were designed in order to study the participant's behaviour.

Secondly, a complete overview of the technologies and how they are used in the actual setup of the experiment.

Then, the desktop application is going to be described from the use of Python to the different libraries and frameworks adopted. Moreover, the main functionalities will be here presented.

The eye-tracking results gotten thanks to the desktop application were processed and analyzed. I will describe the kind of analysis that was done in the postprocessing part : what kind of data was analyzed, what technologies were used and more importantly what are the results that I obtained with this postprocessing part.

Finally, I will present how can be approached a statistical analysis of eye-tracking data, explaining thoroughly what technologies are going to be useful, how can eye-tracking data be used in projects and why it is interesting to adopt a multiple biomarker approach on projects involving the study of a participant's behaviour in response to various stimuli.

Chapter 1 - Introduction to neuromarketing

1.1 Definition of neuromarketing

The neuromarketing field rose thanks to the hypothesis that human feelings, actions and thoughts themselves are solely the product of neural activity in the human brain. Thus, this field consists in studying the brain and to potentially predict or even manipulate consumer behaviour and decision making. The physiological and brain signals are then measured and used in order to get an overview of the customer's way of thinking. After that, advertisement and product development are more adapted to the consumer's needs.

One common method is brain scanning : it can be made through an EEG that is a method to read brain activity by using sensors placed on the subject's head, or this might be done through an fMRI that consists in a non-invasive measure of the blood flow in the participant's brain happening during neural brain activity. It is very expensive to perform and difficult to perform : an fMRI machine costs around $5 million, versus $20.000 for the EEG.

An other way can be the use of biometrics on participants such as the galvanic skin conductance, the heart rate, the respiration. On this project, the team focused on the study of GSR that consists in the phenomenon that the skin momentarily becomes a better conductor of electricity when either external or internal stimuli occur that are physiologically arousing.

One other way of studying the participant's behaviour is by using eye-tracking : we detect precisely where the gaze is directed towards, and we measure precisely the diameter of the pupil, this is called pupilometry. This is the part where I am the most involved.

Finally, we complementarily use Facial Emotion Recognition that is the use of the participant's facial response to the various stimuli performed during this study. It helps to obtain an overview of the emotion response of the subject that is relatively inexpensive.

1.2 The actual experiments

Emotions play an important role in human activity and real-life interactions [3]. There is more and more research on emotion recognition systems that are able to recognize and differentiate people's emotional states. Because emotions contain many nonverbal cues, our study will also analyze various biomarkers as indicators of the participant's emotional states. There are many applications that have been developed with emotion detection as a goal in order to potentially perform mental monitoring, social security.

Here, we focus especially on HCI that plays an important role in recognizing, processing, and detecting a user's emotions. In this project, three different HCI experiments were designed and then studied : Two experiments involving beer brands, and one last experiment that consists in the user to visit a specific website.

The beer brand experiment consists of a research on how the positioning of objects on a shelf or rack can impact the marketing potential of products, in order to improve their consumption. Here we focus on the positioning of beers on a shelf, and we study its impact on the potential consumer. This experiment is applied for the Beer Image experiment. The video experiment consists in introducing beer-related advertisement while watching a TV show and the web browser experiment consists in recording the participant's behaviour while he visits various websites.

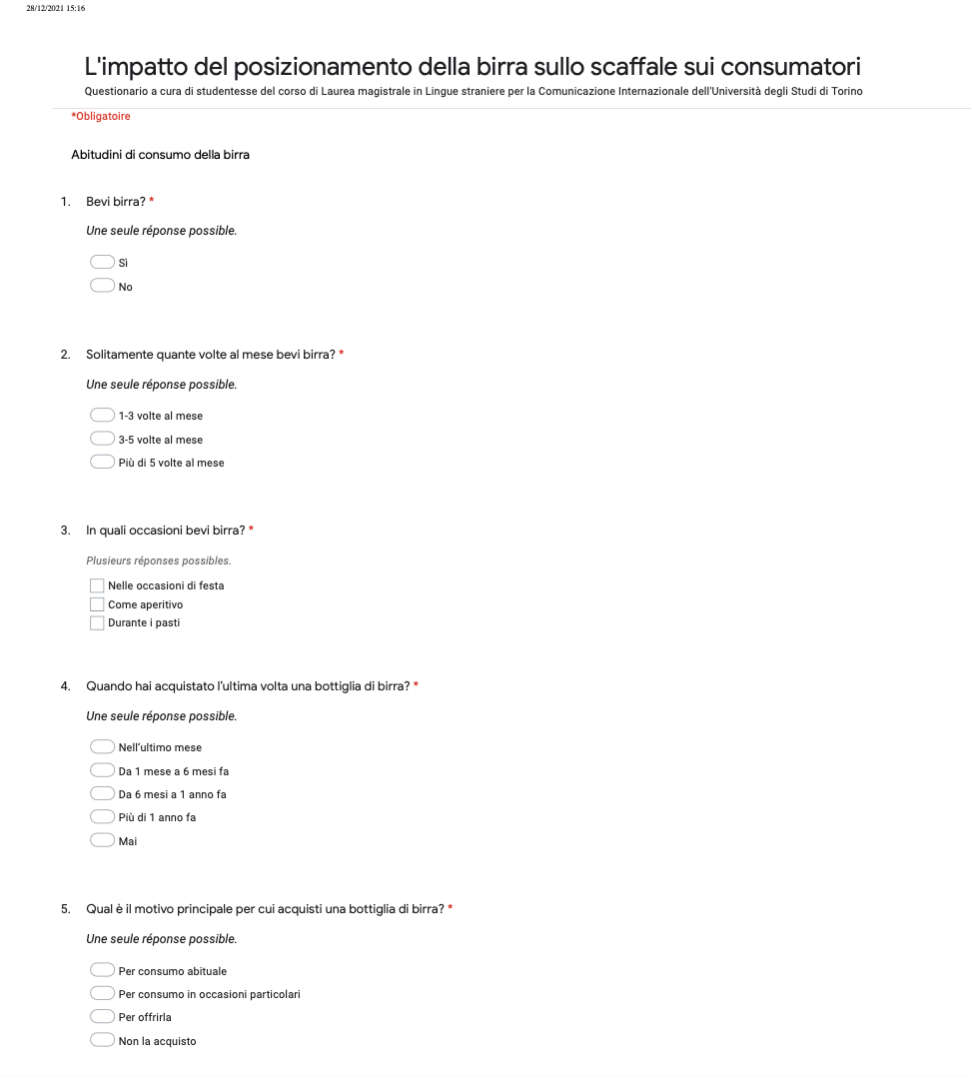
For each scenario, the eye-tracking, GSR and facial encoding outputs are stored for 20 healthy subjects who are enrolled on a voluntary basis. They were divided in two groups with the same number of participants (Group A and

Group B). Each participant filled in and sign an informed consent form prior to the participation in the study.

Laboratory environment:

- A quiet room is chosen to carry out the experiment. All noise from the surrounding environment is set at the minimum possible.
- Lightning condition: we avoid facing direct sunlight, we close the blinds as sunlight contains infrared light that can affect the quality of the eye tracking measurements. We use ambient lights.
- The laboratory temperature is between 23-26°C.
- All system components are placed on a table that does not wobble or shift.
- The eye tracker should be mounted below the computer monitor.
- The webcam should be mounted on the top of the computer monitor.

Figure 1.1 - Laboratory set-up: the image shows the ideal laboratory setup, except for the EEG that is not part of this experiment.



28/12/2021 15:16

L'impatto del posizionamento della birra sullo scaffale sui consumatori

Questionario a cura di studentesse del corso di Laurea magistrale in Lingue straniere per la Comunicazione Internazionale dell'Università degli Studi di Torino

*Obligatoire

Abitudini di consumo della birra

1. Bevi birra? *

Une seule réponse possible.

- Sì
- No

2. Solitamente quante volte al mese bevi birra? *

Une seule réponse possible.

- 1-3 volte al mese
- 3-5 volte al mese
- Più di 5 volte al mese

3. In quali occasioni bevi birra? *

Plusieurs réponses possibles.

- Nelle occasioni di festa
- Come aperitivo
- Durante i pasti

4. Quando hai acquistato l'ultima volta una bottiglia di birra? *

Une seule réponse possible.

- Nell'ultimo mese
- Da 1 mese a 6 mesi fa
- Da 6 mesi a 1 anno fa
- Più di 1 anno fa
- Mai

5. Qual è il motivo principale per cui acquisti una bottiglia di birra? *

Une seule réponse possible.

- Per consumo abituale
- Per consumo in occasioni particolari
- Per offrirla
- Non la acquisto

Figure 1.3 - Extract from the Google Forms questionnary

Different experiments with their actual setup can be chosen on the menu the team designed :

At the end of every experiment, a Google Forms questionnary automatically open so the neuromarketers can get a feedback on how the experiment went, it looks like the below figure : Figure 1.3 (Only the first questions are displayed for clarity purposes).

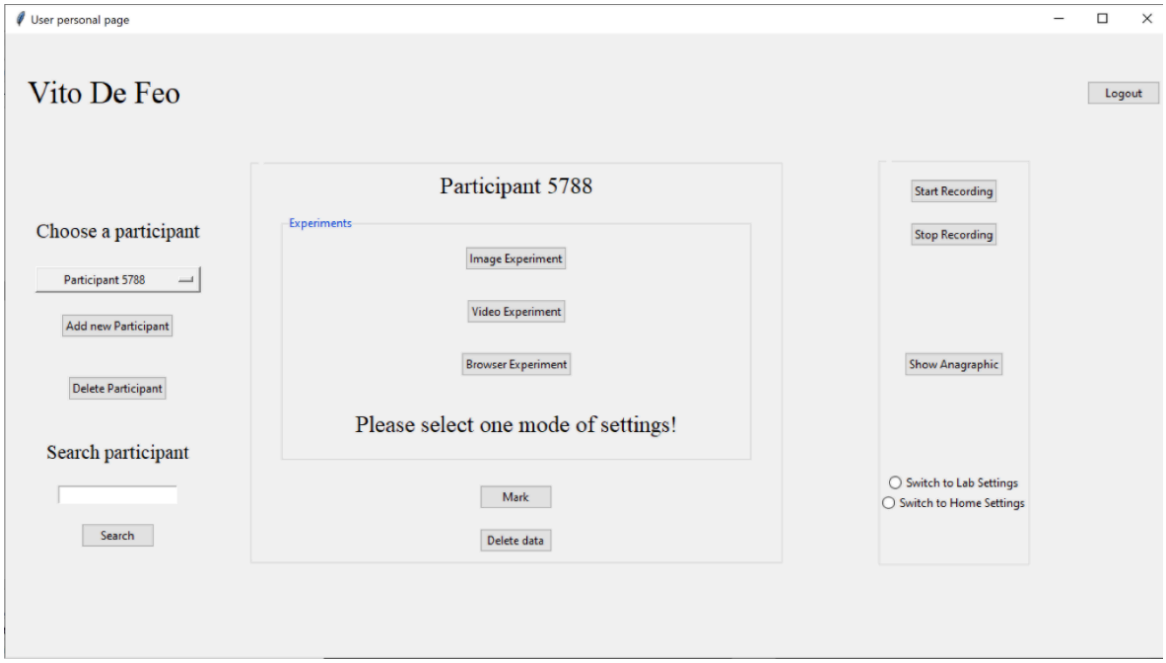


Figure 1.2 - Main page of the Desktop application

1.2.1 Beer Brand Image experiment

Beer images like shown below are shown to the participant for a specific amount of time, two different planograms are displayed in the images clips: two different frames where the beer's position on the shelves is random. The positions used on the shelf follow the logic for how real products are placed on shelves in stores. The two planograms (resolution: 4.8 megapixel) are made up of a sample size of 60 bottles representative of 20 brands. By randomizing the products, it is possible to observe if the amount of attention received by the product and the participant's purchase decision are related to the product position on the shelf. In order to perform the test, two planograms with different product positioning were shown for 20 seconds each to two different groups of participants: Planogram 1- Group A; Planogram 2- Group B.

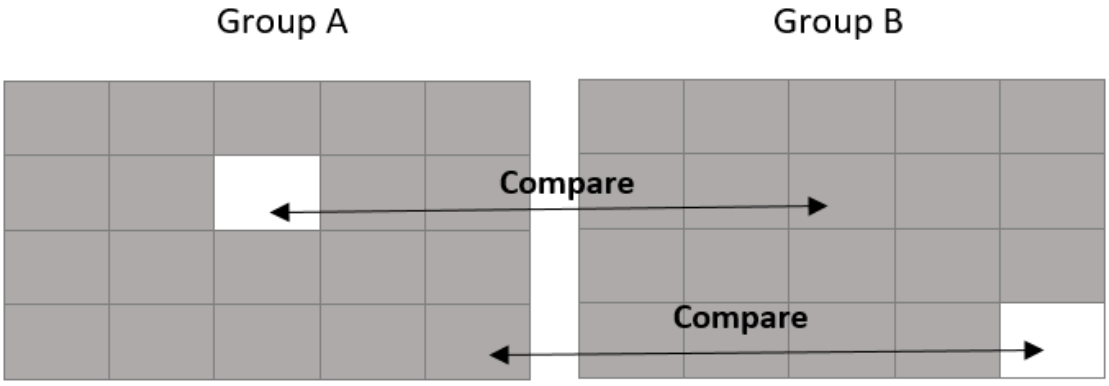


Figure 1.5 - Illustration of the differences between the two different planograms

We implemented the use of a chronometer, an instruction page and also a start recording button. When the participant clicks on the experiment, the instruction page will open and the participant will be able to understand what

Figure - 1.4 Example of image representing the beers on the shelves that are shown to the participants

he needs to do. Then he will click on the start experiment button and the recordings will start along with the displaying of the beer images. All recording are synchronised with the chronometer. When the chronometer stops the experiment will be finished and all the data related to the experiment will be saved in to the participants folder.

1.2.2 Beer Brand Video experiment

The videoclips consist in episode from the show Modern Family where beer ads are included : advertisements for three separate beer brands were used: Ceres, Corona, and Peroni.

After eye-tracker calibration, GSR setup, and video setup the experiment launches, with a dedicated interface adapted to read the episode. When the experiment ends the same questionnaire will open on Google Chrome.

1.2.3 Web browser experiment

The participant accesses pre-defined websites in order to compare two different versions : the actual version of the website and the proposed new version of this same website. After accessing to the website experiment, it is possible to chose between different proposed URLs before launching the experiment. This is the experiment that will be throughly studied on part 4

After that, it is possible to browse freely the selected website for a pre defined amount of time. In the end, all the data related to the experiment is stored on a predefined folder.

Figure 1.6 - Website experiment menu

A navigation bar with 3 different buttons is included with the browser during the experiments : the buttons back, next and reload. Indeed, we realized that the participant might feel lost when he navigates through the website without additionnal buttons and this impacts massively the end results : we

expect the participant to be totally focused on his given task instead of trying to figure out how to effectively browse the given web page.

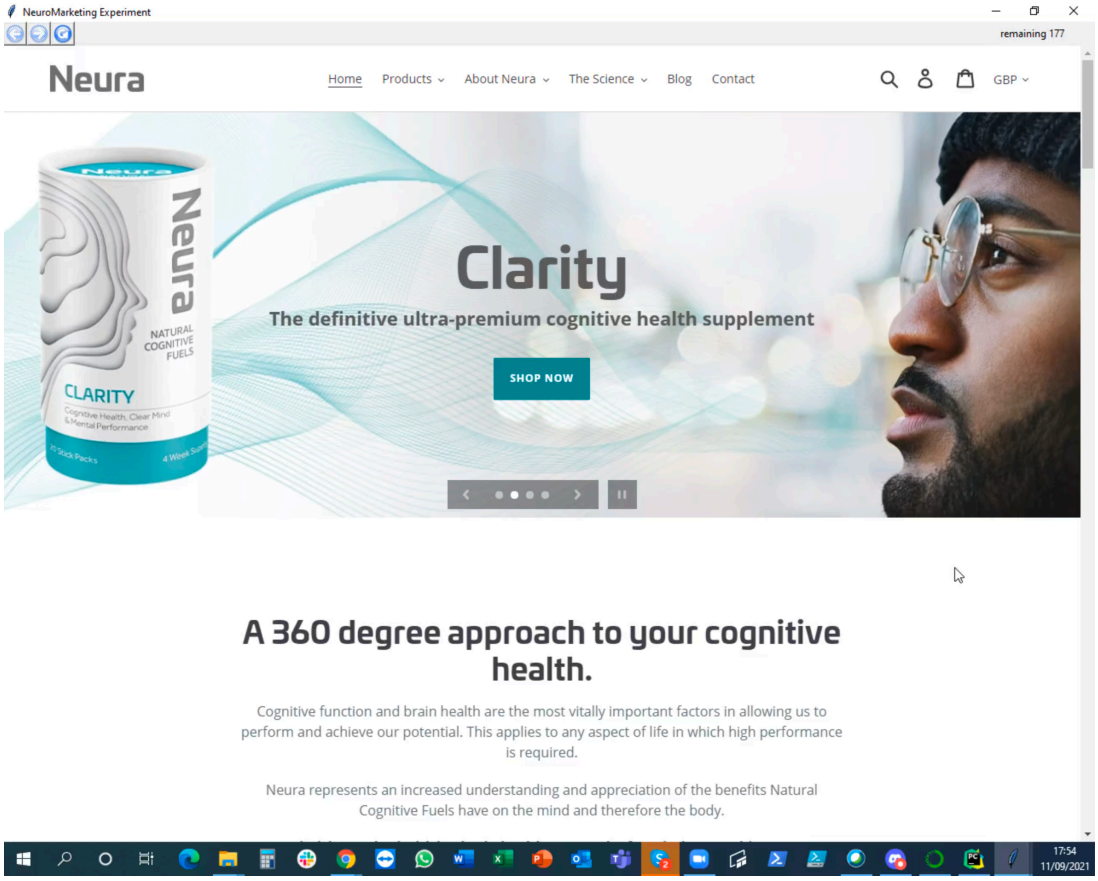


Figure 1.7 - Example frame of the website experiment

1.3 Presentation of the application's functionalities

When the application starts, a login page shows up in order for the abilitated staff to connect to the app safely. After signing in, we can see the main page of the application (Figure 1.2).

There is a wide range of use cases when it comes to visit the functionalities provided by the app from the main page :

- We can Add and delete participants from the local file system.
- Start and stop the webcam recording that was previously used for Face Emotion Recognition (it has been deleted for performance reasons, see Tensorflow's part)
- It is possible to switch between two different sets of settings : Home and Lab settings. As a participant using the desktop application from home, you just have your personnal computer, then only the webcam input, the screen and the sound are recorded during the different experiments. Using the Lab settings, the eye tracker and the GSR eSense recordings are enabled. Using Lab settings, any experiment starts with the calibration of the eye tracker that is mandatory before starting to record.
- When it comes to the experiment themselves, three different experiment possibilities exist : the Image experiment, the Video experiment and the Browser experiment. When the Website experiment starts, a set of instruction is given to the participant. He has to read them before the experiment actually starts.

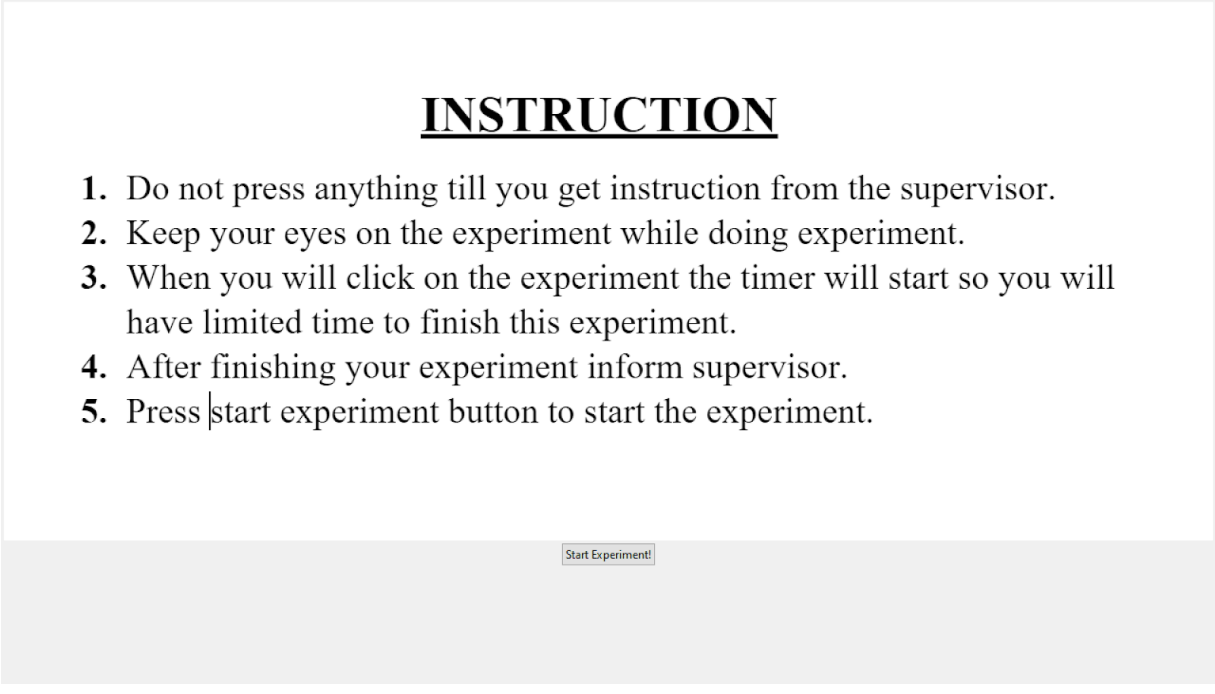


Figure 1.8 - Instruction Frame before an experiment starts

- After selecting a specific experiment item, the recording process starts and the recording data is finally stored when the experiment window is closed.

Chapter 2 - The different technologies used

2.1 Eye Tracking

« Placed on or within the machine interface and using the eye's reflection of near infrared light beams, eye tracking technology calculates data about the user; detecting presence, attention, and focus as well as the position of a person's eye and pupil size. »[4] Eye traDcking data provides unique knowledge about human conduct and human attention while they interact with various types of equipment and devices. Moreover, it helps companies and different platforms to design new systems and products.

This eye tracking technology is used in multiple fields such as neuroscience, neurology, entertainment, marketing, or generally in the study of interfaces between mankind and the machine. The use of eye tracking research is growing too in various health domains : a thorough eye tracking study conclude that the pupillary response time of individuals presenting ASD symptoms is different than heathy participants[5].

Another approach would have been to chose a webcam in order to perform eye tracking, like suggested by C. Meng and X. Zhao in their paper called Webcam-Based Eye Movement Analysis Using CNN [6]. Indeed, the use of an IR eye tracker like the Tobii Pro Nano used for our experiments involve the use of a complex calibration, a higher cost than the use of a webcam and light damage to the eye caused by IR lights. However, a high image quality is required in order to extract pupil, iris and eyelid edges. Thus, both the pupil and glint are often unavailable in videos captured by webcam.

In this experiment, we use eye trackers in order to know exactly where the participant looked on screen, allowing us to determine the most attractive spots on images, videos and on websites. We use Tobii's eye tracking for every

experiment : the Tobii T60 and the Tobii Pro Nano, you can see their pictures below.

Figure 2.1 - Tobii T60

Figure 2.2 - Tobii Pro Nano

The Tobii Pro Nano is a screen-based eye tracker which captures gaze data at 60 Hz and is designed for fixation-based studies. The Tobii T60 contains a monitor and a camera that records the gaze data at 60Hz, it is well adapted to the format of the experiment that we perform : while the supervisor launches the experiment on the screen of the PC, the subject does the experiment using the Tobii T60 monitor.

Using an eye-tracker, we can highlight what is attracting a user's attention and observe some behaviour they might not be conscious of.

2.2 Face Emotion Recognition

Facial Emotion Recognition is a technology used for analysing emotions by using a webcam during the three previously mentioned experiments. It belongs to the family of technologies often referred to as 'affective computing', a multidisciplinary field of research on computer's capabilities to recognise and interpret human emotions and affective states, facial expressions that are a form of non-verbal communicationis used in order to provide clues on the participant's actual emotion.

FER analysis comprises three steps: a) face detection, b) facial expression detection, c) expression classification to an emotional state. [7] These steps will be more detailed on section 5.

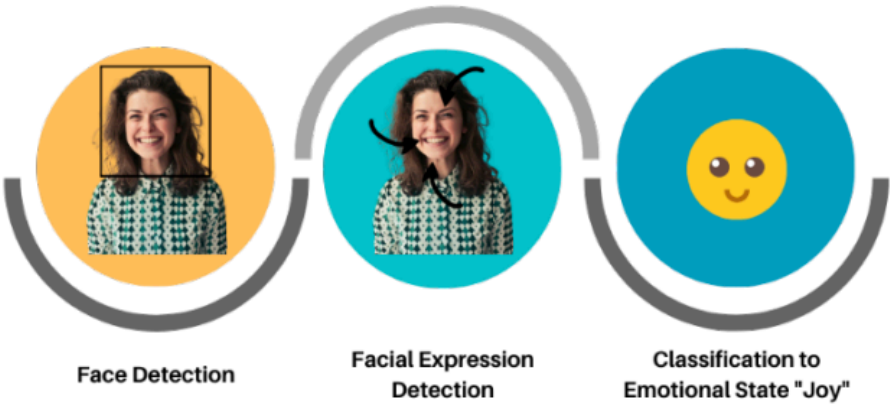


Figure 2.3 - Steps of Facial Emotion Recognition

2.3 Galvanic Skin Resistance

Galvanic skin response which is also known as GSR is a word which refers to differences in activity with the sweat gland which reflect the strength of the emotions, also known as emotional excitement or arousal. When something is terrifying, dangerous, happy or otherwise emotional, our emotional excitement fluctuates in response to our environment, mankind's Sweat Gland activity is increased due to the resulting emotional shift. Positive (happy or happy) as well as negative (threats or disappointment) incentives can increase the arousal and hence the behaviour of the skin. The GSR signal indicates emotional intensity.

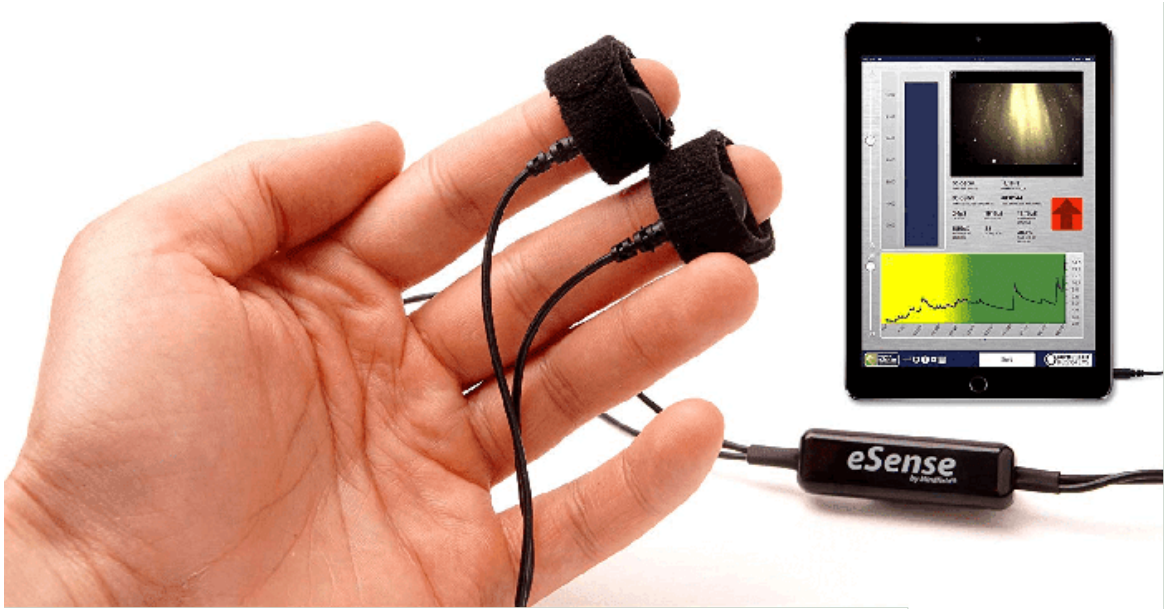


Figure 2.5 - eSense GSR equipment

GSR tests must be dynamic and be able to send the electro specific information to a recording device, since it can detect variations in the electrical activity produced by changes in sweat glands activity[8]. The most advanced GSR electrodes touch the skin through Ag/AgCl (silver chloride). Ag/AgCl conductors are used because they are cost-effective, robust and safe for the human touch and can precisely transmit the signal from electrical activity. In order to increase signal fidelity. Ionic gel is pre-applied to some electrodes or ionic gel is injected to the very same result. In each scenario, the signal is sent from the cathode to the wire, which generally delivers the information to the GSR equipment. From here, the information is either kept for further uploading on the device, given to the computer network wirelessly or converted into digital through a wired connection. Various GSR sensors are capable the various transmission modes and the sort of study you are undertaking determines the one you pick. Leading is assessed with skin electrodes that are easy to apply. The data is measured in micro-Siemens and the sampling rate applied is between 1 and 10 Hz.

Chapter 3 - The desktop application

3.1 Python

Python is a high-level language of programming developed in the early 1990s by Guido Van Rossum. The language is pseudo-interpreted, which is an interpreter switches on a virtual machine and runs one code line by line, converting this into a machine language that agnosticizes the platform. A range of programmes such as procedure, imperative, functional and objective-oriented programming objects are supported by Python, which enable reasonably flexible use of language to be adjusted to the programmer's demands and skills. It may also be used for orders that only share the running sequence (scripting), but also to create sophisticated, object-oriented platforms that are totally cooperative [9].

Another major aspect of the language is the strong dynamic typing, which permits the allocation of any kind of value to variables, enabling more versatility in programme design but only the risk of errors being detected in runtimes, when everything seemed to operate properly in advance. Lastly, Python uses a waste collector predicated on the theory of production and barrier of objects, which indicate the number of objects in a particular generation, when a 'collection process' is started to free the memory of components not already in use in the application.

Lastly, SDKs for all of the other tools, beginning with the face and proceeding through the eye trackes, coupled with the GSR, require Python as a programming language. As Python is platform-independent, there are still no problems with the environment it is used, therefore we can begin working on Microsoft without that much concern for compatibility.

It was then agreed by the thesis team project to work on Python and to develop the app using this language, which makes the app compatible with both the CNNs and the different SDKs required to run the embedded programs

in the eye trackers and the GSR eSense. The version of Python 3.6 was chosen for compatibility purpose with the Tobii eye tracker.

3.2 The different libraries used

I built the virtual environment using Anaconda and a dedicated .yml environment file as you can see on the image below. Anaconda is a distribution of the Python that is dedicated to data science or machine learning applications[10]. This distribution aims to simplify the package management and deployment inside the project, which was essential because we needed to work in a coherent virtual environment. The main goal of this Anaconda configuration was to simply recreate the same environment for every member of the team project in order to not face configuration errors.

```
name: psychopy
channels:
- conda-forge
dependencies:
- python=3.6
- psychopy=2020.2.4
- pip
- pip:
  - psychtoolbox
  - pyglet==1.4.10
  - pyo
  - pyparallel; platform_system != "Windows"
  - SoundFile; platform_system == "Windows"
  - websocket_client
  - python-vlc
  - cefpython3
  - tobii-research
  - pygame
  - screeninfo
  - tensorflow
```

Figure 3.1 - The .yml environment file

By using this configuration file and a dedicated GitHub repository, we could eliminate most of the issues related to configuration of a project involving multiple members working on the same project.

3.2.1 Tkinter

« The tkinter package (“Tk interface”) is the standard Python interface to the Tcl/Tk GUI toolkit. Both Tk and tkinter are available on most Unix platforms, including macOS, as well as on Windows systems. »[11]

Tkinter was chosen in order to provide the experiment staff a consistent GUI that is indeed simple but it contains all the necessary logic for the project.

It is possible to embed all the three performed experiments in the Tkinter GUI, as well as the main menu, the login page, and the participant creation section.

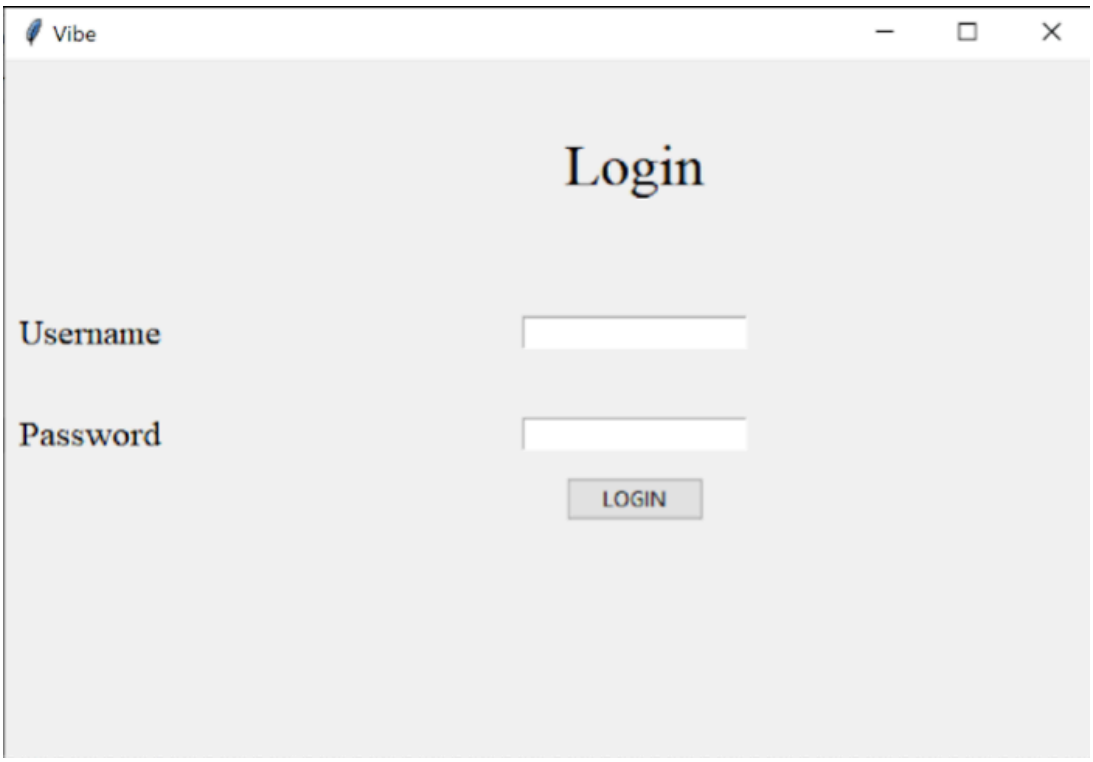


Figure 3.2 - Example page using Tkinter (Login Page)

Tkinter has the advantage of provinding an efficient yet simple interface that is understandable by the staff experimenting at the Caserta Lab. One other

advantage using Tkinter is that the various experiments' pathway becomes very clear according to the neuromarketings' set of instruction when we use the Tkinter parent/child windows.

For example in this case, top is the parent window that is then used to create a video frame, which is created for the video experiment :

```
class Player(Tk.Frame):
    """The main window has to deal with events.
    """

    def __init__(self, parent, frame_=None, title=None, type=None, path=None, sec=None):
        Tk.Frame.__init__(self, parent)
```

Figure 3.3 - Player class using Tk.Frame instanciated as a child

```
def createVideoFrame():
    top = tk.Toplevel()
    top.title("Experiment VLC media player")
    top.state('zoomed')
    player = None
    player = vp.Player(top, title="tkinter vlc")
```

Figure 3.4 - Example use case of the previously shown Tkinter class, with top as the parent frame

3.2.2 Psychopy

Psychopy is the most important library used in the project. PsychoPy is a free cross-platform package allowing us to run a wide range of experiments in any domain concerning behavioural sciences.[12]

It can be directly used as an application through the Psychopy Builder Interface, or using a programming language such as Python. The second option was chosen, in order to possibly custom our experiments using our Tkinter GUI.

This library is totally platform independant, allows us to use a wide variety of stimuli like texts, images, sounds, videos.

The most important feature is that this library is compatible with a huge range of experiment hardware especially the Tobii eye trackers, eSense GSR measure tool, webcams we use in this project. This easy communication was essential to the aims and requirements of the project, which relies significantly on the registration of biomarkers for many of its structures.

Moreover, the use of multiple monitors is possible and this feature is used for the experiment so that the staff can supervise the experiment on its own screen while the participant does the experiment on his side.

The relevant thing to keep in mind is the fact that Psychopy or any experiment framework is really useful for the experiment's coordination feature that is implemented : this library allows us to perform experiments using multiple sensors in a synchronyzed way. Experiments can be received only if we can keep track of time for every different physiological signal by using timestamps for example, which is the case in this project.

Then, routines can be implemented using Psychopy. Every routine specifies a scenario particular to the subject and which it should react to; inside the routine, components that indicate different sorts of stimuli or activities that should be permitted or recorded are entered from the right menu. The procedures are the various stages of a test. Each element has multiple parameters that may tailored to the desires of the study designer with easy-to-use options, simplifying considerably the screen creation presented to the participants. Finally, each routine should be put in the right sequence to produce the study flow desired by the investigator.

One must recognize that the workflow desired by the investigator is not always the most optimals so many times adjustments and compromises were made to meet the psychologists' requirements..

After the experiment is finished, the app produces pure Python code which may be modified in the Coder to a lower level, a basic text editor that is flexible for non programmers to use.

3.2.3 Tobii-research and Titta

Using the Tobii Pro SDK is mandatory in order to be able to use any Tobii eye tracker. In Python, Tobii-research wraps the Tobii Pro SDK so Tobii-research is mandatory to use in Python for our project. The SDK contains the required drivers for the use of any eye tracker, and in the event of a stolen eye tracker, the Tobii Pro license is blocked so that the eye tracker is not usable anymore for reasearch.

This particular library allows the user to connect fastly to any Tobii eye tracker connected to the computer, to perform a calibration and to store data as a dictionnary. The main limitation comes when we want to use the eye tracker complementarily with PsychoPy : we don't have an adapted toolbox for using in an optimal way the eye tracker functions, especially when it comes to calibration. This is the reason why we are using Titta : an open-source toolbox for controlling eye trackers manufactured by Tobii AB from MATLAB and Python.[13] This is a wrapper to the Tobii Pro SDK that uses fully Psychopy experimental functions. This wrapper particularly includes a calibration interface that is easily customizable and dedicated to any type of user, may they be code developers or not.

Then, in order to fully use the Titta configuration, one just needs to import the library, to use the required settings by the experiments (all the experiments we performed only needed default settings). After that, we create a window that corresponds to the monitor used to show the stimuli, and use

this monitor for the calibration. Finally, when the calibration is done, we can start the recording and launch our experiment. You can see all these steps in Figure 3.5.

```
settings = Titta.get_defaults(et_name)
settings.FILENAME = 'data/Browser/' + str(participantId) + '/' + data.getDateStr() + '.tsv'

print(settings.FILENAME)

settings.N_CAL_TARGETS = 3

tracker = Titta.Connect(settings)

if dummy_mode:
    tracker.set_dummy_mode()
tracker.init()

# Window set-up (this color will be used for calibration)
win = visual.Window(monitor=mon, fullscr=FULLSCREEN,
                    screen=1, size=SCREEN_RES, units='deg')
fixation_point = helpers.MyDot2(win)

tracker.calibrate(win)
win.close()

tracker.start_recording(gaze_data=True, store_data=True)
webInstruction.launch_browser(search_key_var, type, participantId, parent, root, frame)

tracker.stop_recording(gaze_data=True)
tracker.save_data(mon)  # Also save screen geometry from the monitor object
```

Figure 3.5 - Example Python code for using Titta on the Web browser experiment

In the end, we obtained a user friendly graphical interface embedded in the Psychopy experiment that allows the experiment staff to proceed to the eye tracking calibration and to record data on a dataframe that is easily converted to a .csv file afterwards. Moreover, we can generate a very useful dataframe that provides us the timestamps corresponding to each phase of the calibration and to the starting moment of the experiment, allowing us to exactly know when the experiment occurs regarding the generated data.

The calibration can be customized : we can choose any number of calibration targets we deem necessary for the experiment and it is possible to previsualize the calibration points in order to evaluate the calibration that was

previously performed and to accept it, or to restart it if the calibration points are not precise enough.

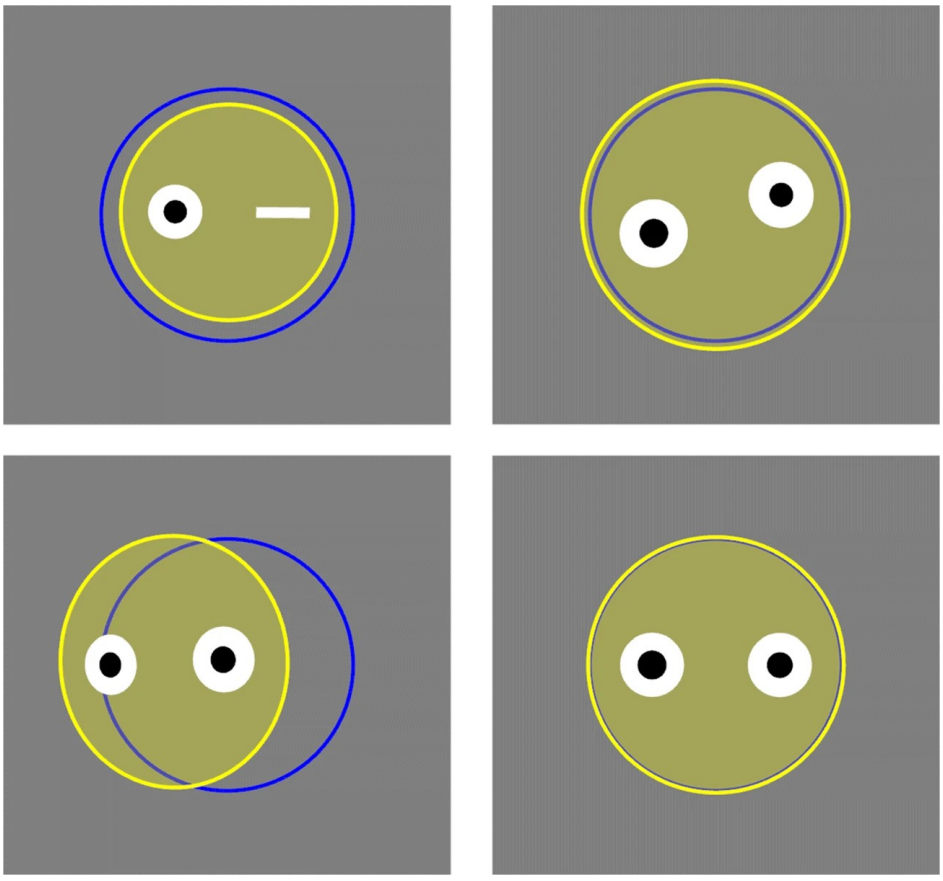


Figure 3.6 - beginning of the calibration, when the participant is asked to fill his head in the blue circle

3.2.4 Other relevant libraries

A set of Python libraries were required in order to fufill the project's requirements. Cefpython3 and python-vlc were used to respectively embed the web Browser experiment and the video experiment :

Cefpython3 is an open source project founded by Czarek Tomczak in 2012 to provide Python bindings for the Chromium Embedded Framework (CEF)[14]. This library is used in order to create a Synchronized

Google Chrome browser that is embedded inside our Tkinter window, that allows the participant of the experiment to browse a website freely during the experiment. The Python command for the synchronized browser can be seen on the Figure 3.7 below.

```
def embed_browser(self):
    window_info = cef.WindowInfo()
    rect = [0, 0, self.winfo_width(), self.winfo_height()]
    window_info.SetAsChild(self.get_window_handle(), rect)
    self.browser = cef.CreateBrowserSync(window_info,
                                         url=self.starting_url)
    assert self.browser
    self.browser.SetClientHandler(LoadHandler(self))
    self.browser.SetClientHandler(FocusHandler(self))
    self.message_loop_work()
```

Figure 3.7 - Synchronized browser creation using CefPython3

This library serves as a HTML5 based rendering engine that can act as a replacement for classic desktop GUI frameworks. Besides, logs are present so that we can assess in real time the functioning of the browser. However, one big drawback that exists using this library is that this framework does allow the main Python thread to perform Javascript callback functions that would have been really helpful to record the scrolling position at any time. What had to be done instead was to record the screen using FFMPEG and to calculate at any moment of the experiment the scrolling position on the website using the scrollbar as a referential (see Postprocessing part).

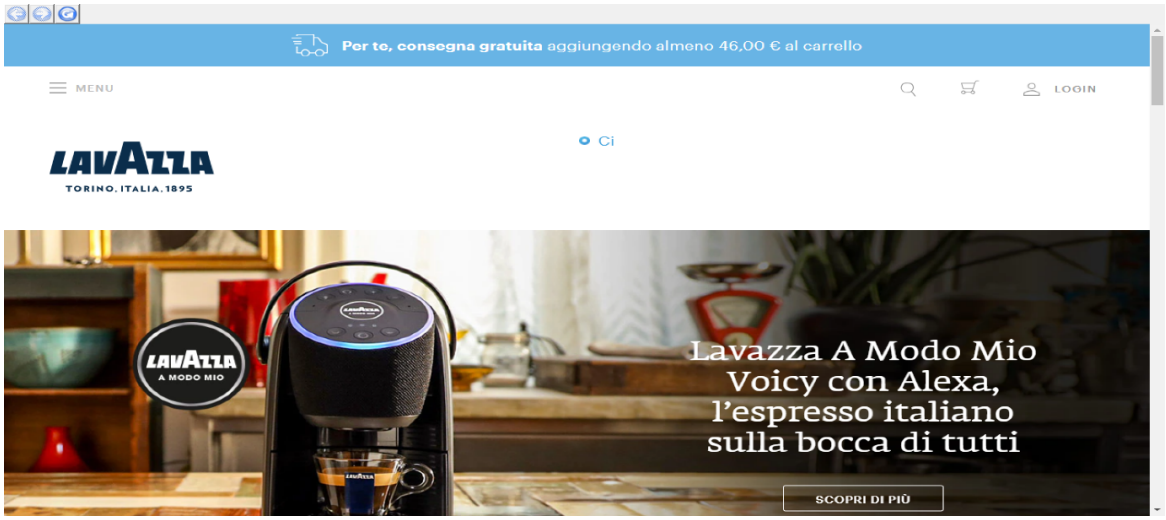


Figure 3.8 - Example of Browser experiment snippet

Python-vlc : The Python-VLC multimedia player is a portable player which is capable to play a broad array of video and music formats. recently, the focus has been on video streaming in diverse types. This library has a modular approach so that you can only use some functions if you are interested in different output kinds or formats. LibVLC is the key engine of the VLC Media Player, together with the external software platform. You should be able to use this module to obtain the same functionality as your original player. The reason we chose using python-vlc instead of Psychopy directly is the following : a video of around 20 minutes with acceptable quality needed to be used for one of the neuromarketing studies mentioned previously. The duration of this video led to conflicts between its audio and video chronology, as visual control in Psychopy is not really optimal : it was not designed for experiments that last for such a long time. As a consequence we choose to construct a multimedia player in our application using the Python-VLC module.

Figure 3.9 - Example Frame from the Video experiment

One other relevant library to mention is FFMPEG : FFmpeg is a command prompt application capable of encoding and decoding a series of types of media. A full cross-platform audio and video recording, conversion, and streaming solution. We use it to record both the experiment screen and the webcam pointed towards the face of the participant, with a 60Hz refresh rate.

Tensorflow was used for applying the previously mentioned Face Emotion Recognition CNN. Because of the fact that the app was significantly slowed down while using tensorflow and all the other modules at the same time, it was decided to not keep it even if it was up and running. Indeed, multiple functions were already functioning simulteanously and this particular one made many crashes occur. Instead, an external cloud based application was created where the previously mentioned CNN was functioning.

Finally, the design of the app was made through a multithreading approach. Python multithreading allows us to execute several parts of our programme at once, facilitating design. This means multiple processes will happen simultaneously in your software. Nevertheless, the separate threads don't actually work at the very same time throughout most Python 3

applications; they just seem to. At first, we didn't use a multi-threading approach to design the software and we had too many concurrent operations like video recording, GSR, face emotions recognition, experiment running at the same time. As a consequence, the technology lagged badly. In order to resolve this difficulty, we had to begin creating functions from the ground, and utilised a multithreading approach. Threading provides numerous simultaneous processes that considerably help resolve the lagging problem, even though we were forced to rethink the architecture of the whole application at the moment we thought was near to be finished.

Chapter 4 - Postprocessing

4.1 The goal of postprocessing

The postprocessing part consists in the analysis of the experiments that were performed using the app, especially the web Browser experiment. This part especially was involved in the comparison of two versions of the same website : the old version and the mockup version of the website https://www.neuracognition.com (that became the used version since). 14 different participants were asked to browse during 3 minutes each version of the website, and their behaviour was recorded using the Desktop application. Here, the focus was set in eye tracking so we mainly used the Tobii eye tracking data obtained by the Desktop Application using the Lab Mode and the screen recording of the application.

For the two studied versions of the website, we are comparing the participant's behaviour when he browses three different pages : the main page, one of the product's page, the product description for the newer version and the main page, one of the product's page and the science behind the products for the old version of the website. (See Annex for the website images)

Python was chosen for this part of the project : it goes in the same vein as the Desktop application and, more importantly, OpenCV exists in Python. Besides, data processing can be performed using Pandas and data visualisation can be done using seaborn so Python was adapted for every side of the postprocessing side of the project.

As mentioned earlier, the scrolling of the web browser could not be recorded using CefPython3 so we had to find a way to record the scrolling of a participant using a particular page of the website. This is the reason why OpenCV was used coupled with the screen record in order to obtain the scroll position at any time.

4.2 The technologies used

4.2.1 Obtaining scrolling position

OpenCV (Open-Source Computer Vision Library) is a free and open-source software library for computer vision and machine learning. OpenCV was created to provide a common infrastructure for computer vision applications and to help commercial goods incorporate machine perception more quickly. [15]

On this particular context, the OpenCV library was very useful because I needed an algorithm that is able to detect quite simple objects like a scrollbar. I used the Haar Cascade classifiers[16] in this particular use.

Each frame of the screen recording undergoes some preprocessing before the scrollbar can be recognized :

- I used a GaussianBlur to the image : it is used for reducing the amount of noise present in each frame.
- Then, a thresholding function is used : we transform using pixel thresholds a color image to an adapted black and white image, customized for the scrollbar recognition. You can see an example frame resulting from the thresholding function on Figure 4.1.
- Finally, as the scrollbar is in the right side of the screen, we decide to only detect objects where the scrollbar is : at the right side of the screen.

You can see the result of preprocessing in the below image where the scrollbar is quite visible in black and white. The final result can be seen Figure 4.2, where the scrollbar is enlightened in yellow after the use of all the mentioned steps on preprocessing the frame using OpenCV.

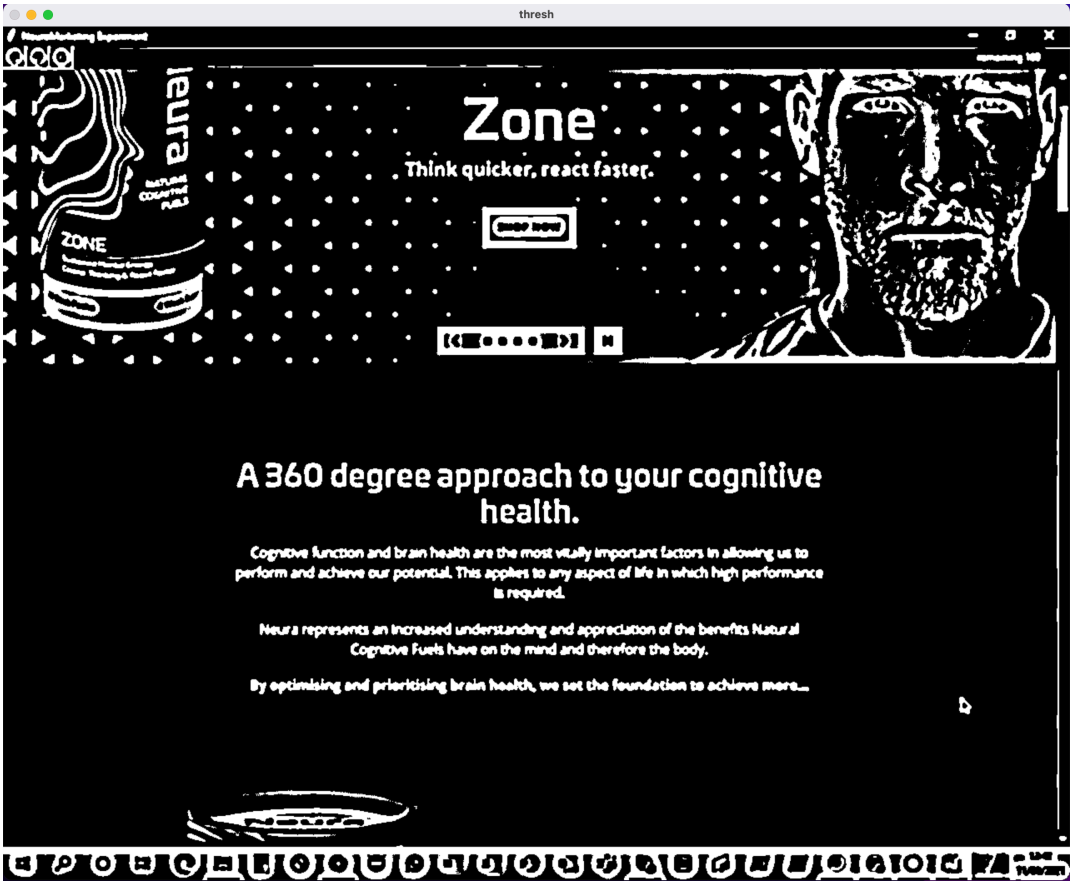


Figure 4.1 - Example frame after applying GaussianBlur and threshold to a frame

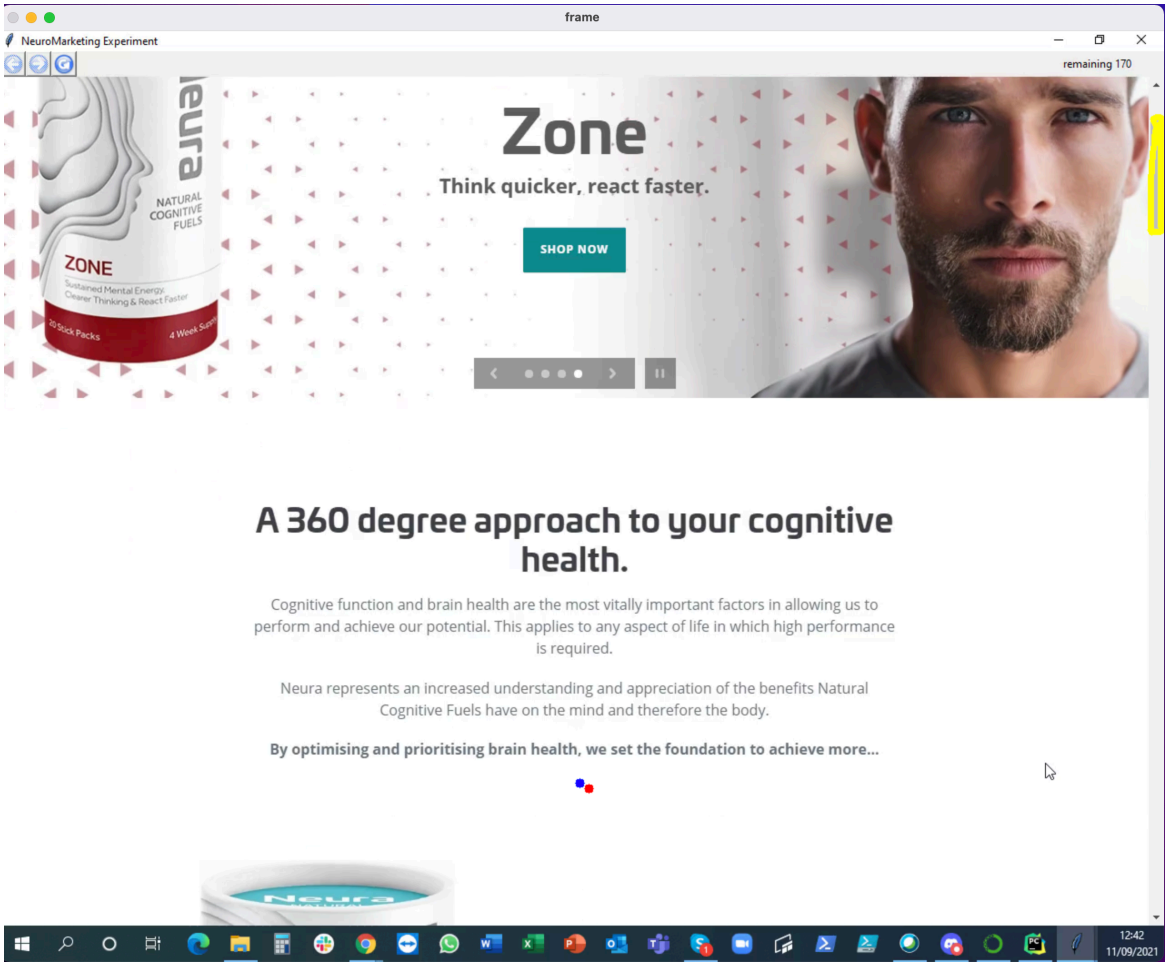


Figure 4.2 - Example final frame after the application of all the preprocessing methods using OpenCV

An implementation of these functions can be seen on Figure 4.3 and Figure 4.4 where I used a while loop in order to perform the same operations on all frames from a specific screen recording. We use the different filtering operations on the image : the Gaussian blur and the thresholding, xl, yl, xr, yr correspond to the x,y position of the eyes (respectively left and right eye). Finally, the openCV function findContours is used to obtain the position of the scrolling bar

```
while cap.isOpened():
    ret, frame = cap.read()

    if ret == True:
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

        img_blur = cv2.GaussianBlur(gray, (9, 9), 0)
        thresh = cv2.adaptiveThreshold(img_blur, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY_INV, 23, 3)

        xl = int(new_df.loc[index, 'left_gaze_point_on_display_area_x'] * width)
        yl = int(new_df.loc[index, 'left_gaze_point_on_display_area_y'] * height)

        xr = int(new_df.loc[index, 'right_gaze_point_on_display_area_x'] * width)
        yr = int(new_df.loc[index, 'right_gaze_point_on_display_area_y'] * height)

        index += 1

        center_coordinatesl = (xl, yl)
        center_coordinatesr = (xr, yr)

        radius = 5
        # Red color in BGR
        redcolor = (0, 0, 255)
        # blue color in BGR
        bluecolor = (255, 0, 0)
        # Line thickness of -1 px
        # Using cv2.circle() method
        thickness = -1
        # Draw a circle of red color of thickness -1 px
        frame = cv2.circle(frame, center_coordinatesl, radius, redcolor, thickness)
        frame = cv2.circle(frame, center_coordinatesr, radius, bluecolor, thickness)
```

Figure 4.3 - Implementation of the scrollbar retrieving function

```
contours, _ = cv2.findContours(thresh, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
for idx in range(len(contours)):
    x, y, w, h = cv2.boundingRect(contours[idx])
```

Figure 4.4 -Command used to detect the scrollbar by using the Haarscascade classifier

Both the websites we are studying, the new and the old version, do not require a lateral scrollbar, which removes one parameter to take in account. When the scrolling bar's position on the website is determined, we can use it in order to calculate the scrolling position on the website.

Selenium is an easy-to-use Chrome and Firefox extension and is generally the most efficient way to develop test cases. It records the users' actions in the browser, using existing Selenium commands, with parameters defined by the context of that element [17]. Using Selenium as an extension for Google Chrome on Python, we can determine the actual size of each web page in pixels.

Indeed, if we call :

- amp the amplitude between the scrollbar's upper position and its lower position
- height the height of the scroll bar
- startPos the starting position of the scrollbar
- y is the position of the scrolling bar on screen

By hypothesis, the scrolling position of the scrolling bar is linear when compared to the actual scrolling position. Then, following a linear relationship, we obtain that the formula for the scrolling position is :

$$ScrollPos = (height/amp) * y - (startPos * height/diff)$$

The same calculus has been done for the three different pages that were studied for the old and for the new version of the website. Besides, we can automatically determine in which page of the website we are located by using the size of the scrollbar, that is inherent to each page.

Moreover, it is possible to see on each frame a blue and a red dot : they correspond to the participant's left and right eye. I drew them using OpenCV and the data stored in a csv file that I obtained from the desktop application.

The tobii eye tracking's sampling rate is 60Hz, and the screen recording's refreshing rate is 60Hz : this is a really important point because synchronization is mandatory if we want accurate results.

As a result, using the scrollposition and the positioning of the eyes, we can determine at any moment where is the gaze point on the website. The resulting data is stored in a Pandas dataframe.

4.2.2 Data creation and storing

Pandas is an open source and reliable solution designed for creating and storing dataframes. It is widely used in Data science and dataframes are highly reliable when it comes to manipulate large amounts of data. Besides, this library has been optimized for performance when it comes to manipulate huge sets of data, which is of use in this project.[18]

In order to determine Areas of Interest for further analysis, I decided to separate each image as a grid, containing an arbitrary number of rows and columns. The more rows and columns this grid has, the more accurate the Areas of Interest are, but the results are less readable, especially for the heatmaps.

This is the reason why each page is separated in a grid containing 30 rows and 8 columns : on the different Figures 4.5, Figure 4.6 and Figure 4.7 you can see a comparison between different scan paths coming from the same experiment, with a different set of parameters concerning the grid : 10 rows and 5 columns for Figure 4.5, 30 rows and 8 columns for Figure 4.6, 100 rows and 50 columns for Figure 4.7. Those three different scan paths represent the same experiment, but with different rows and column parameters.

While the postprocessing program is running, each row is concatenated to the pandas Dataframe corresponding to the right page. Using pandas, I finally store on an excel file a fixation matrix reporting the duration of the fixation on each page of the website. I am using this matrix in order to draw heatmaps corresponding to the participant's gaze points on the website.

Because of the fact that 3 different pages from the website are studied, one fixation matrix is drawn for each page.

The number of rows and columns of each fixation matrix is equal to the number of rows and columns on the heatmaps. The duration of the fixation on each part of the website is shown in blue, you can see an example of a heatmap I drew in the Figure 4.8.

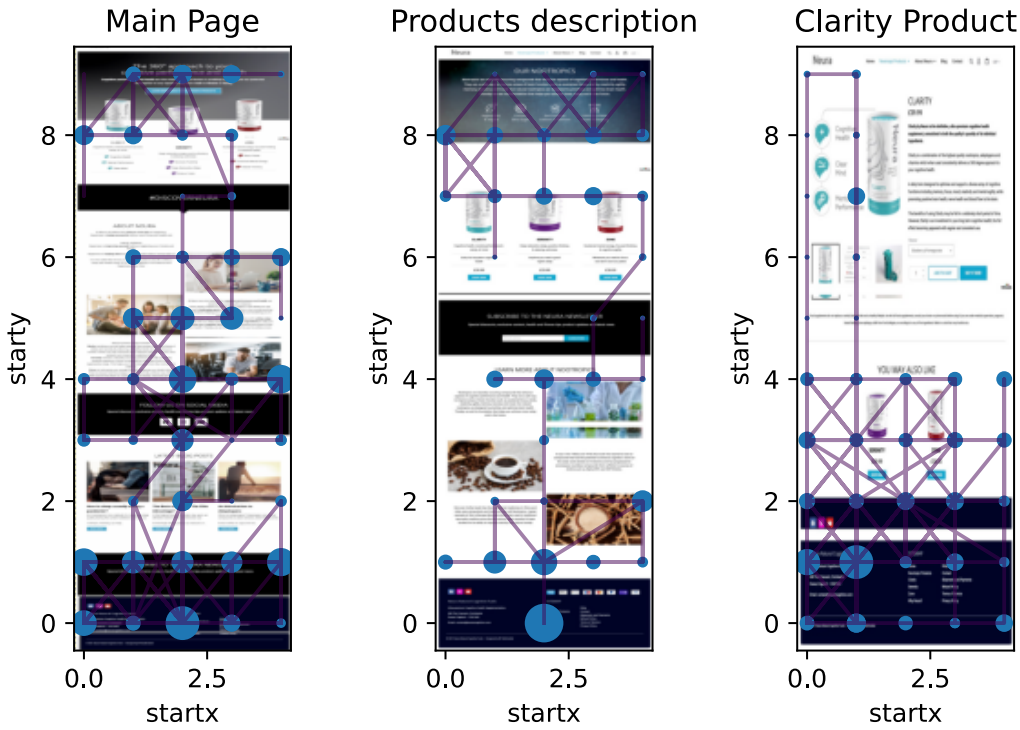


Figure 4.5 - Scan path with values 10 and 5

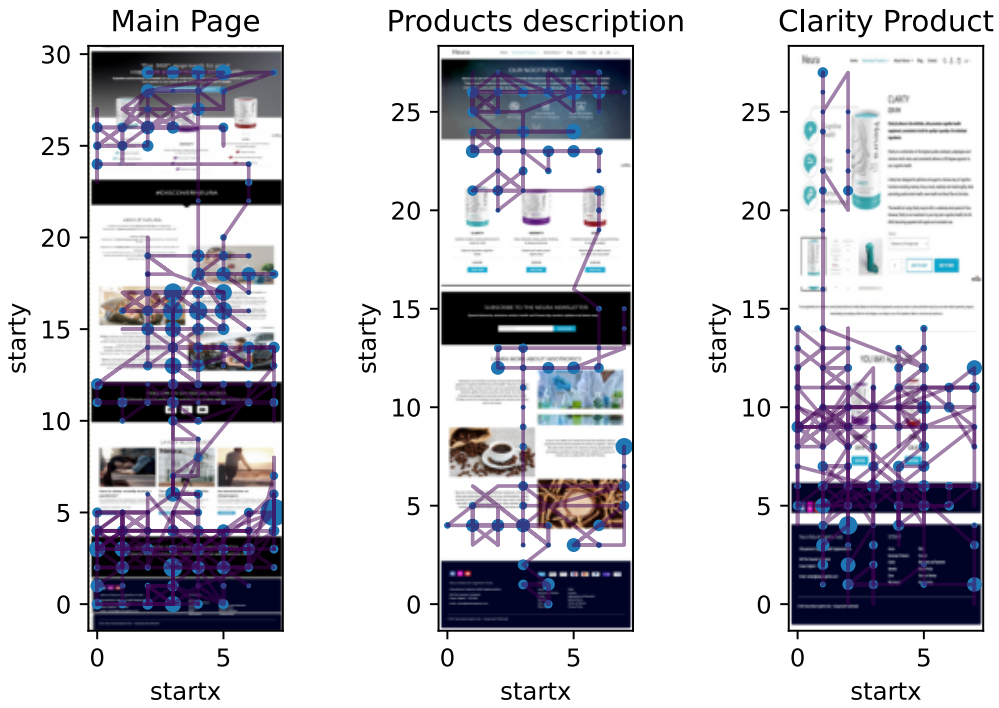


Figure 4.6 - Scan path with values 30 and 8

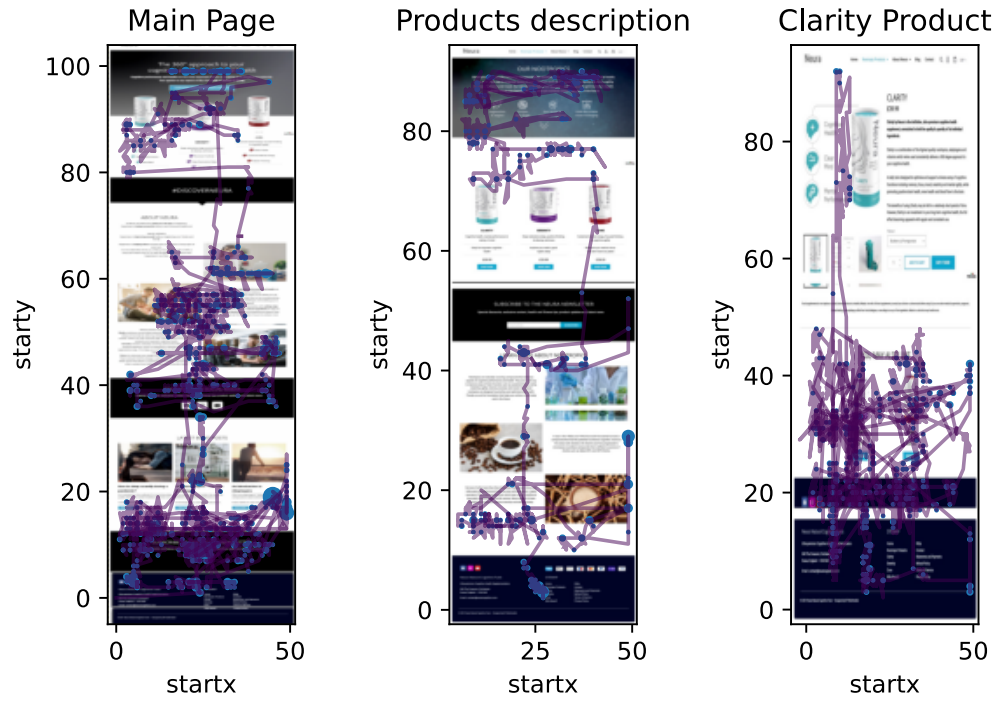


Figure 4.7 - Scan path with values 100 and 50

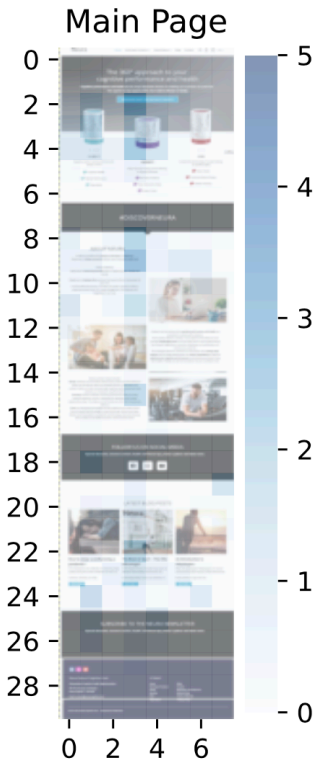


Figure 4.8 - Example of heatmap drown for the Website's main page using 30 and 8 as row and column parameters

Another Dataframe is created in the same way while the postprocessing program is running : its goal is to create a scan path. A scanpath corresponds to eye-movement data collected by a gaze-tracking device, where information is recorded about the trajectories (paths) of the eyes when scanning the visual field and viewing and analysing any kind of visual information. Such data usually consists of gaze-direction, fixation position and duration. In order to draw one scan path, we need the coordinates of their eyes plus the duration of each fixation. For readability purposes, we link the different fixation points using lines instead of arrows. Figures 4.5, 4.6 and 4.7 are examples of scan paths using different parameters.

4.2.3 Visualization

The heatmaps were created using seaborn. Seaborn is a Python data visualization library based on matplotlib[19]. It provides a high-level interface for drawing attractive and informative statistical graphics. This library was chosen for its possibility to obtain a graphical view of the data provided by the multiple experiments done so far.

Matplotlib plots were used in order to create the scan paths : they consist in linking the different gaze points and to secondly creating a scatter plot corresponding to the duration of each fixation : the bigger the dot is, the longer the fixation was (see Figure 4.5, Figure 4.6, Figure 4.7).

For both result types (heatmaps or scan paths) the web page image that was seen by the participant during the experiment using the desktop application was overlapped with the heatmap (or the scan path) so that we can obtain a visual result regarding where did the participant look.

You can see how the heatmap is obtained on Figure 4.9, where the image coming from the website (main_page, clarity_science or clarity_page) is overlapped with the corresponding heatmap. The same method was used with matplotlib.pyplot for creating the scan paths.

4.3 The results obtained

For neuromarketing purposes, the different participants all answered a questionnaire at the end of the website experiment. This questionnaire helped the neuromarketing specialists distinguish the participants into various categories :

- Love for surprise (typically womanlike) regarding people who tend to be more easily surprised by what they see in everyday life. This leads to a greater interest in those things.
- Attitude to mistrust (typically manlike), regarding people who, on the contrary of “love for surprise”, tend to be more reluctant to appreciate something.

```
map_img_main = mpimg.imread('Images/website_images/main_page.png')
map_img_products = mpimg.imread('Images/website_images/clarity_science.png')
map_img_clarity = mpimg.imread('Images/website_images/clarity_page.png')
#map_img_main = mpimg.imread('Images/mockup_images/main_page.png')
#map_img_products = mpimg.imread('Images/mockup_images/products_description.png')
#map_img_clarity = mpimg.imread('Images/mockup_images/clarity_page.png')

fig, axes = plt.subplots(1, 3)

vmax = max(main_matrix.max(), products_matrix.max(), clarity_matrix.max())
ax1 = sns.heatmap(main_matrix, vmin=0, vmax=vmax, ax=axes[0], alpha=0.50, cmap="Blues", zorder=2)
ax1.imshow(map_img_main,
           aspect=ax1.get_aspect(),
           extent=ax1.get_xlim() + ax1.get_ylim(),
           zorder=1)  # put the map under the heatmap
# ax1.collections[0].set_alpha(0)

ax2 = sns.heatmap(products_matrix, vmin=0, vmax=vmax, ax=axes[1], alpha=0.50, cmap="Blues", zorder=2)
ax2.imshow(map_img_products,
           aspect=ax2.get_aspect(),
           extent=ax2.get_xlim() + ax2.get_ylim(),
           zorder=1)
# ax2.collections[0].set_alpha(0)

ax3 = sns.heatmap(clarity_matrix, vmin=0, vmax=vmax, ax=axes[2], alpha=0.50, cmap="Blues", zorder=2)
ax3.imshow(map_img_clarity,
           aspect=ax3.get_aspect(),
           extent=ax3.get_xlim() + ax3.get_ylim(),
           zorder=1)
# ax3.collections[0].set_alpha(0)
```

Figure 4.9 - Implementation of heatmap using seaborn

- Interest in nootropics (specifically related to the experiment), regarding people who have been particularly attracted by nootropics and the way they works on the cognition and the body.
- Attitude to life, regarding people who think they need to improve their life at best.
- Sports interest, a specific factor regarding people who tend to be attracted by sport supplements.

Beyond those categories, we kept the gender categories (male or female) and then aggregated the fixation matrixes regarding the amount of time spent on each region for each category. For example the Figure 4.10 represents the heatmap corresponding to all the people belonging to the « Love for surprise » category.

Using aggregated heatmaps for a group of person allows to obtain relevant trends regarding the time spent on each region of the website.

According to the heatmaps that were created, participants seem to deepen the observation more on the new version compared to the old one. The heatmaps show gaze points and fixation points broader and longer on the mockup, especially on the written parts, where they were given information about nootropics (the product sold by the website), their composition or why they ought to be taken. A possible explanation is that people looked less at the original because there was more text than in the new version.

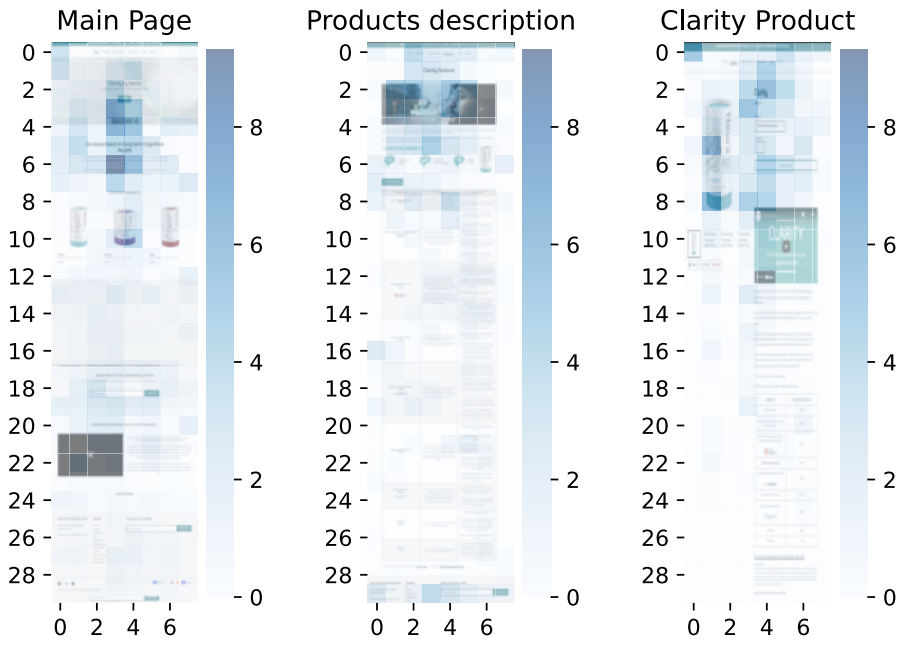


Figure 4.10 - Aggregated heatmap for the« Love For Surprise » category

Specific AOIs were obtained using the squares coming from the design of the app. For instance, using the 30x8 format, 2x6 squares were used to describe the AOI corresponding to the main illustration on the products description page (see Figure 4.7). Using this system, we could determine the fixation time for each AOI.

On the old version people focused most on the images presented on the mainpage (pop-up images) - average total fixations at 12,62 seconds, and on the "Science page" - 9,7 seconds, but not on the most important information, such as products description or details about how they work. On the contrary, the new version presents two main AOIs: first of all, the Serenity product presents an average fixation of 10,45 s - Clarity stops at 3,91s and Zone at 3,01s. Regarding the "Science" page, again, people diminished attention towards the products even if people seemed to like the detailed explanation on what are nootropics. Regarding the last page, for both sites, did not show any important AOI to take into account, but it seems that on the mockup people spent a little bit more time: 1,22 seconds for every fixation compared to the 1,10 seconds for every fixation on the original site.

The questionnaire was used in order to proceed to the comparison of the olv and the new version of the website : for the question "Do you think Neura improves general well being?", there is a statistical significance in the asnwer($z = -2.226$; $p = .026$), in favour of the original site – without an important trend. In other words, this sample can confirm that a specific population looks at the original site as a little bit more able to improve the general wellbeing, but not so much to justify a clear trend in its favour.

Chapter 5 - Statistical approach of the study of emotions

5.1 Prediction using eye tracking

This first part will essentially describe how is eye tracking data obtained using an eye tracker such as the one used in our experiment : the Tobii Pro Nano or the Tobii T60.

Eye tracking and gaze estimation are two different fields of research. The process of eye tracking mainly involves three different steps :

- Obtaining the input image, generally by using a webcam or an eye tracker that is for the moment raw data.
- The presence of eyes is determined, and a quite precise interpretation of eye positions is made by generally measuring the pupil center.
- The position of the eye is Gaze estimation is a process to estimate and track the 3D line of sight of a person, or simply, where a person is looking. This process is made by the eye tracker itself. A gaze tracker performs two main tasks simultaneously: localization of the eye position in the video or images, and tracking its motion to determine the gaze direction. A generic representation of such techniques is shown in Figure 5.1 [20]

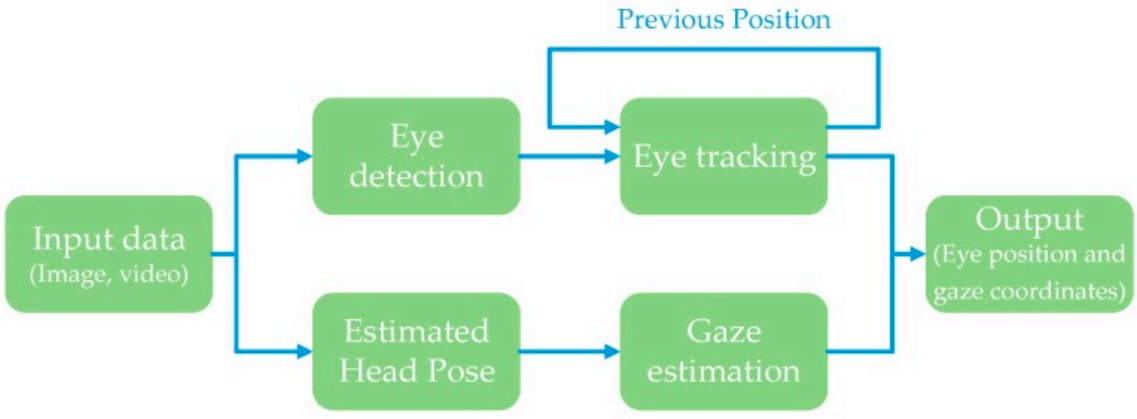


Figure 5.1 - The process of tracking eye position and gaze coordinates.

Let's proceed with eye detection : the detection of the eyes in an image or in a video input is based on the different existing human eye models. The method adopted by Tobii eye trackers normally works under variable lightning conditions and facial orientations, using the active IR illumination approach : The pupil that is exposed to IR light is utilized to simultaneously detect and track the eyes. Pattern-classification recognition is obtained through the use of SVM, while object tracking is obtained through the mean shift method.

Various parameters such as the degree of eye openness, variability in eye size, the head pose, and eye reflectivity are mandatory to take in account for eye detection and tracking. For instance, a small variation in viewing angle or head position causes significant changes in the eye appearance or gaze direction, as shown in Figure 5.2. The eye's appearance is also influenced by ethnicity of the subject, light conditions, texture, iris position within eye socket, and the eye status (open or closed). Eye detection methods are broadly categorized based on eyes' shape, features, and appearance, as explained below. [20]

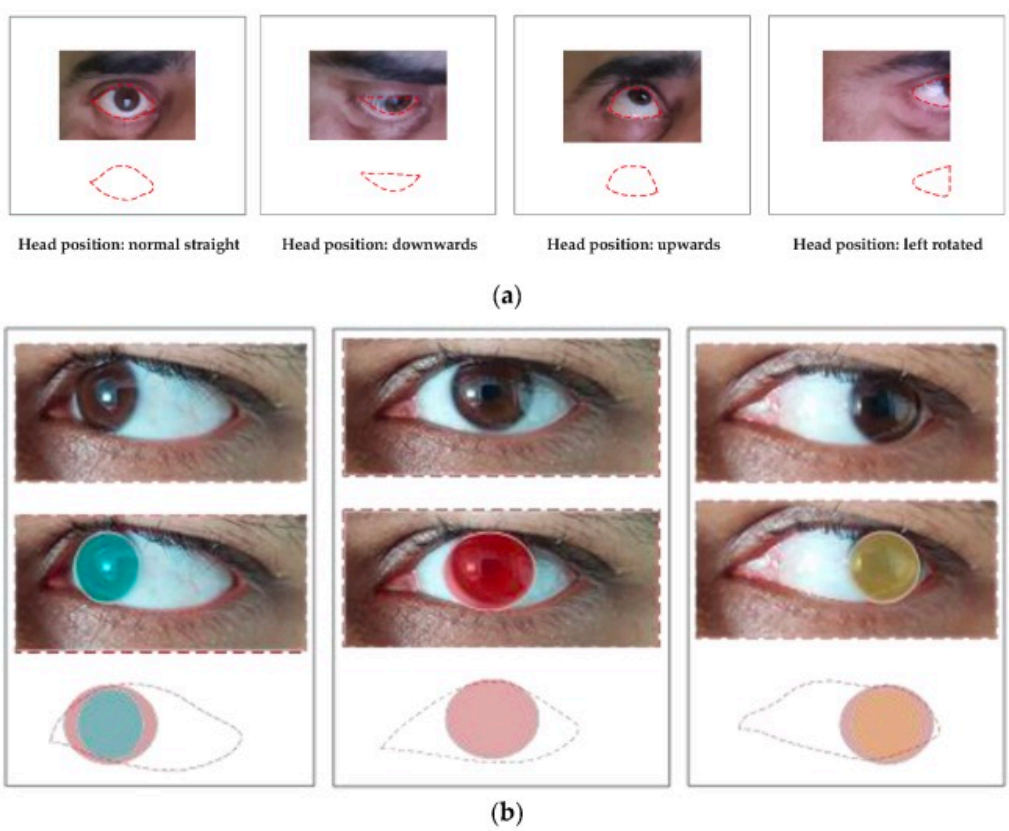


Figure 5.2 - Detection of the eyes in different positions, from the head to the eyes

In the end, we obtain the following flowchart that describes how eye tracking is performed for most eye tracking devices (Figure 5.3):

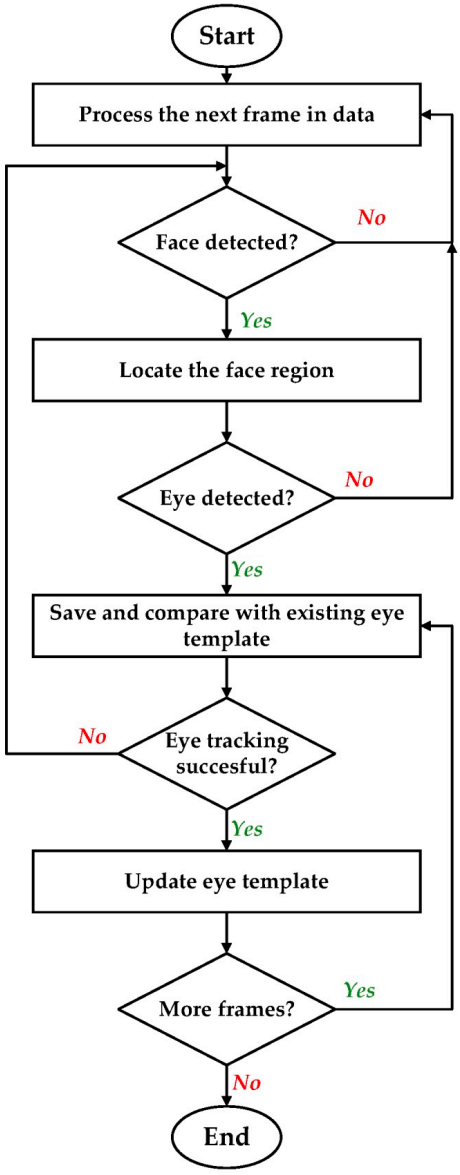


Figure 5.3 - Flowchart of a generic eye tracking algorithm

Using Tobii eye trackers and their Python libraries, a number of information regarding the eyes of participants during any experiment coud be recorded in real time, especially time stamps, gaze points and eye availability. The availability corresponds to the fact the eye could be recorded : for instance,

at the moment when the participant blinks, the availability of the blinking eye (or both eyes) comes from 1 to 0. An empty row in any record corresponds to a local error in the recording, there are less than 1% of them in the recording files, so this is not significant in the obtained results.

After this, the eye tracking data that we could gather should be analyzed for emotion recognition purpose for example. Using the postptrocessing application that I designed, heatmaps and scan paths could be drawn. Besides, we could calculate fixation duration on specific AOIs. However, the postprocessing app is limited when it comes to the design of machine learning techniques adapted to emotion recognition.

Interesting features can be extracted to recognize emotions including blinks, fixation duration and saccade duration. Using these features, multiple approaches can be imagined for studying emotions such as neural networks, Bayesian networks, unsupervised, semi-supervised, and supervised machine learning techniques for further analyses.

There are already existing softwares for eye tracking data analyses available as open source projects like PyTrack [21]. It is an analysis tool kit available either as a GUI or as a Python project that is meant to analysing and visualizing eye-tracking data. It can be used to extract parameters of interest, generate and visualize a variety of gaze plots from raw eye-tracking data, and conduct statistical analysis between stimuli conditions and subject groups. Moreover, custom AOIs can be drawn using their User Interface and the time spent on each AOI automatically obtained.

The following Figure 5.4 explains how are experiment subjects of a specific experiment processed for further statistical analysis using PyTrack.

This framework also provides the functionality of performing statistical tests such as the *t* test and variants of ANOVA for combinations of between and within group parameters, that would be very helpful for a more thorough study

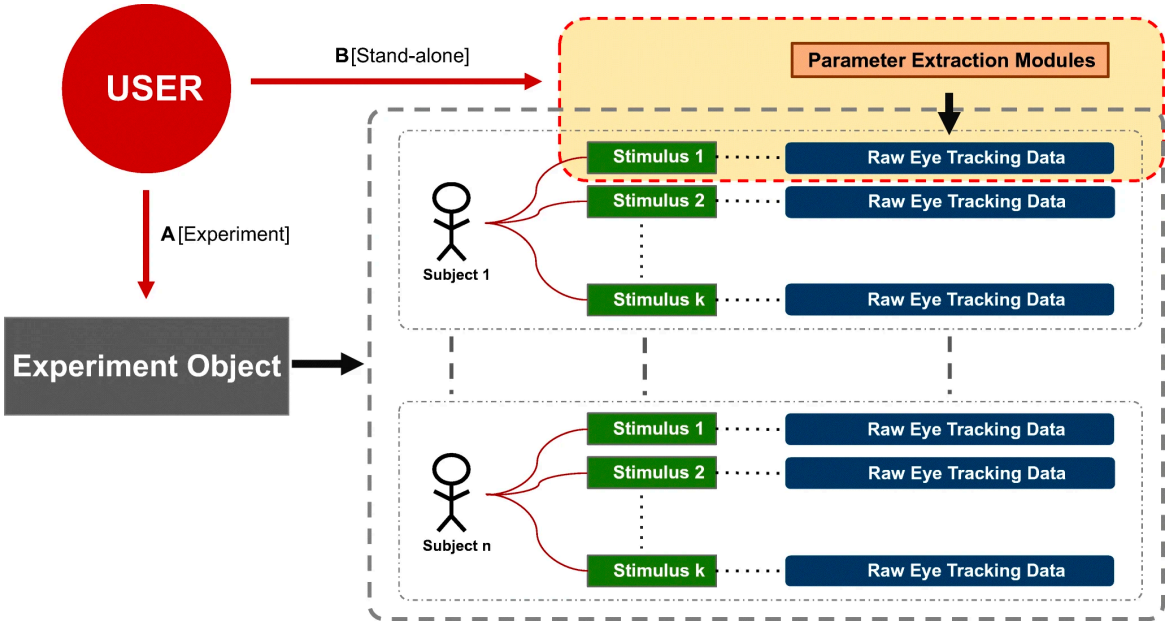


Figure 5.4 - PyTrack framework structure

of the obtained eye tracking data through the different experiments. Besides, the Tobii eye tracking data is supported by this particular framework that makes it be a very interesting tool for eye tracking's data study.

However, the eye tracking data has to be slightly adapted for possibly using this particular software : PyTrack is adapted for performing statistic analysis for data coming from static images, whereas the statistical analysis that we needed to perform was on a website, consisting of multiple pages which size is superior than the size of an image, so scrolling must be taken in account. So, the open source PyTrack software should be adapted, this could be the starting point for further analysis using the posptrocessing software.

5.2 Prediction using Face Emotion Recognition

Humans can experience a wide range of emotions, that were separated in 7 different categories for being able to perform classification : happy, sad, angry, disgusted, frightened, startled, and neutral. We decided to then perform classification for one of these 7 emotions, using an existing model based on arousal and valence that can be seen on Figure 0.1 : the Circumplex Model of

Affect [2]. Using this model, each emotion can be placed on a quadrant driven by arousal and valence.

The AfectNet [22] was chosen to train the CNN model for emotion classification as it was an emotion based dataset using valence and arousal.

FER analysis is made on three steps: a) face detection, b) facial expression detection, c) expression classification to an emotional state.

Face detection : There are numerous algorithms for recognising faces, objects, classifying human activities, tracking moving things, and creating three-dimensionalmodels of items, OpenCV is one of them.

This library contains a number of other very useful functions especially on computer vision and machine learning algorithms, as it is used for the designed app and for postprocessing purposes. This is why OpenCV was chosen to perform face detection.

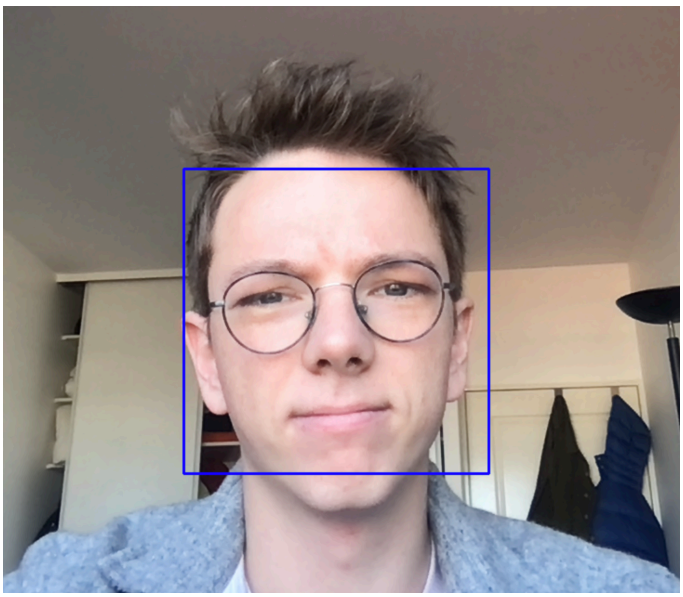


Figure 5.5 - Face detection using OpenCV

Facial expression detection : A VGG16 Neural network model was chosen to perform feature extraction. VGG16 is a convolutional neural network proposed by K. Simonyan and A. Zisserman of Oxford University and gained notoriety by winning the ImageNet Large Scale Visual Recognition Challenges (ILSVRC) in 2014. The model achieved an accuracy of 92.7% on Imagenet which

is one of the highest scores achieved. It marked an improvement over previous models by proposing smaller convolution kernels (3×3) in the convolution layers than had previously been done. The convolution kernels 3x3 have a stride equal to 1 and always use the same maxpool layer 2x2 with a stride equal to 2. Throughout the architecture, the convolution layer and maxpool layer combination is repeated.

As this model is used for feature extraction, the two fully connected layers you can see on the figure below (Figure 5.5) were replaced by a 1x512 numpy array meant to serve as an input for future predictions.

This CNN is probably the most adopted architecture for facial soft biometrics analysis. Indeed, the availability of weights pre-trained on a very large number of face images and not for general image classification task (ImageNet) makes it very suited for transfer learning.

To increase the efficiency of our VGG16 model, transfer learning methods were applied using an ImageNet dataset of more than 14millions images. Its structure can be seen on Figure 5.6.

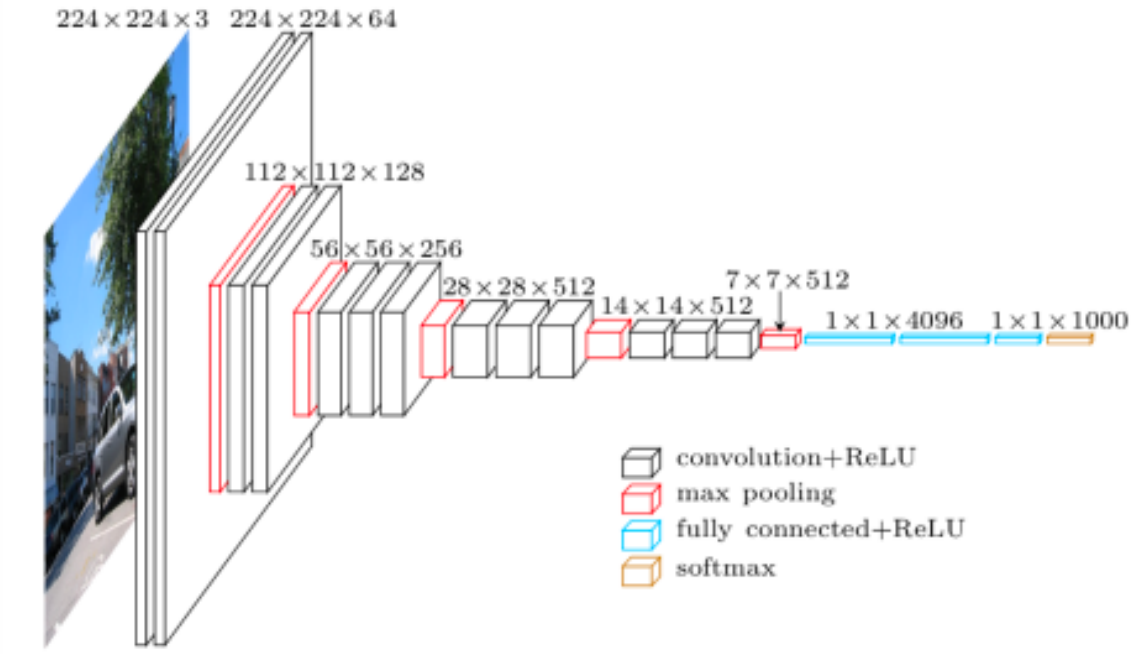


Figure 5.6 - Structure of VGG16

expression classification : The prediction model is made up of three layers that are all fully coupled. After experimenting with many models of varying complexity, this combination was chosen.Because predicting arousal and valence in two-dimensional space might be difficult, the model should be as efficient as possible while keeping the overall structure's execution time to a minimum. The experiment led to a prediction model that can be shown in figure 5.7 after taking into account all of the criteria related to accuracy and execution time.

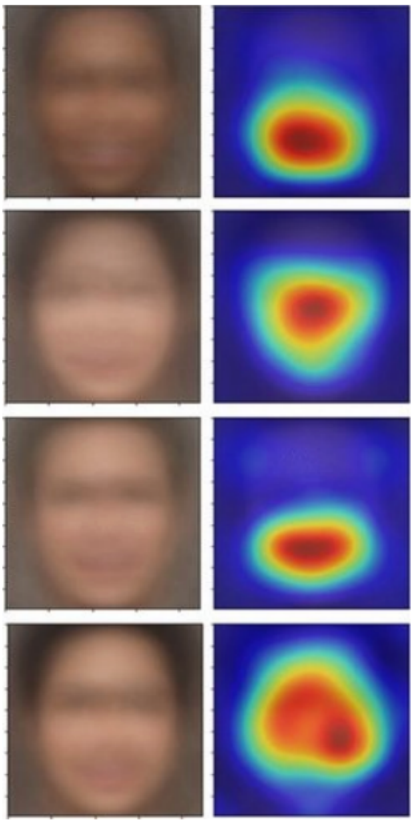


Figure 5.7 - Average face images and class activation maps obtained by applying VGG16

The mean squared error loss function is used in the training phase of this model (MSE). The loss function is represented by the equation below. Assume that Y is the right output that the network should return (also known as Target),

and that Yˆ is the output that the network actually returns. The average loss function for all cases in the training dataset will be:

$$\mathbb{E}_D[(Y - \hat{Y})^2]$$

The model is trying to adjust the probability distribution of Yˆ such that it equals the probability distribution of Y . Indeed, the model is trying to make Y = Yˆ , such that the mean squared error becomes 0. The following equation shows the lowest possible value of the mean squared error:

$$\mathbb{E}_D[(Y - \hat{Y})^2] \geq 0$$

Knowing that the variance of Y is always positive, then the mean loss-function has the following lower-bounds:

$$\mathbb{E}_D[(Y - \hat{Y})^2] \geq Var(Y) \geq 0$$

The training approach used several LR (Learning rates) (0.0001, 0.001, 0.01), optimizers, activation functions, and dropouts to optimise hyperparameters that were used to train the model (0.25, 0.5). In addition, two alternative activation functions, Softmax and Tanh, are used in the final layer. MSE was chosen as a loss function in the back propagation function

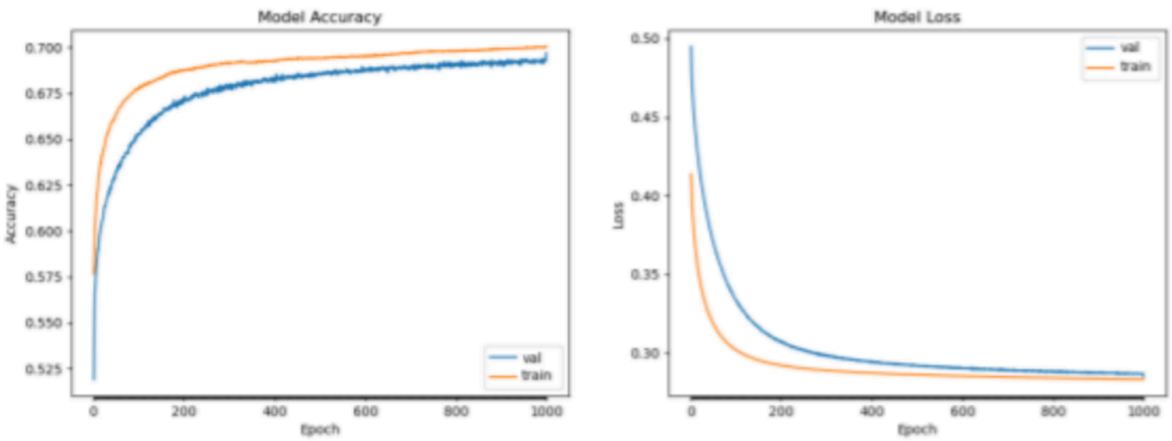


Figure 5.8 - The accuracy (left) and loss (right) on our custom CNN model

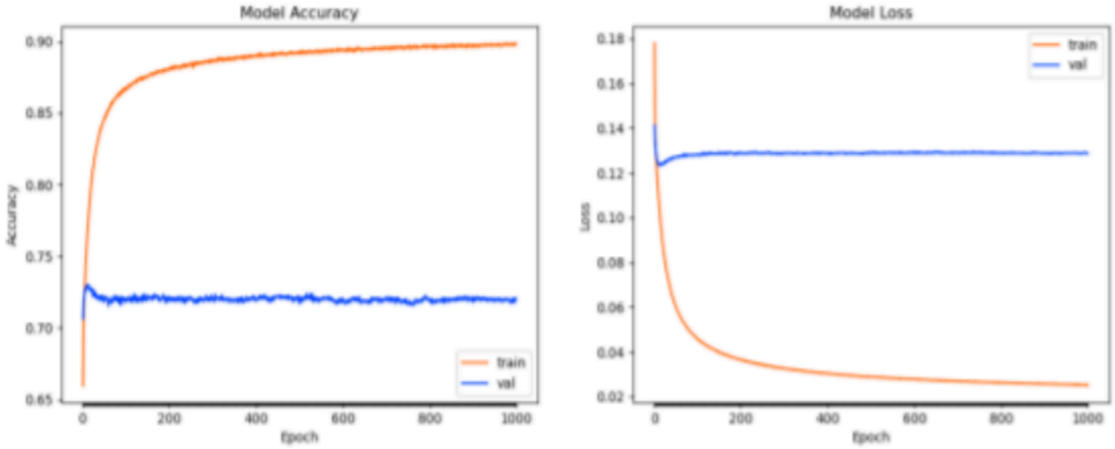


Figure 5.9 - The accuracy (left) and loss (right) on our custom CNN model with modified fully connected layers

5.3 Prediction using a multiple biomarkers approach

The paper from Zheng, Wei-Long & Dong, Bo-Nan & Lu, Bao-Liang [23] is a great introduction to the potential in involving multiple biomarkers in order to perform this kind of study on participant's emotion : EEG and eye tracking were used simultaneously on a study involving participant's emotions. The best accuracies of the EEG and eye tracking models when used alone were 71.77% and 58.90%, respectively. However, when used on their fusion model, an average accuracy of 73.59% is obtained.

The final goal of this project is to create a way to detect emotions using multiple biomarkers : the GSR, face emotion recognition and eye tracking. It is indeed intuitive to think that using multiple biomarkers is going to be more effective than using a single one. However, the features from each biomarker must be wisely chosen in order to avoid overfitting issues : for instance, it has been discovered that the pupil diameter changes according to the emotion state the participant is in. Then, keeping the pupil diameter as a feature for a fusion CNN involving multiple parameters looks like a wise choice.

The more biomarkers are chosen for a multiple biomarker approach, the more complex the final classifier is going to be, even if the use of a PCA is still possible in order to select the most relevant feature adapted to your study[24]. On both multi biomarkers approaches seen, a feature based fusion model was used.

The method described in this paper[25] is called a feature fusion technique .This method was applied for emotion recognition performed on a valence-arousal emotion space by using hidden Markov models (HMMs) and multimodal feature sets.

On the left side of the Feature-level fusion in the following Figure 5.10, feature extraction methods are applied on four modalities individually to obtain modality features. Then, features extracted from various biomarkers are selected and combined to be a lower dimensional feature set for recognizing the emotion and a higher dimensional feature set for recognizing both dimensional feature sets that are identified in the training phase. The multimodal features used in the testing phase are entirely the same as the significant features identified during the training phase; therefore, feature selection algorithms are not reused in the testing phase.

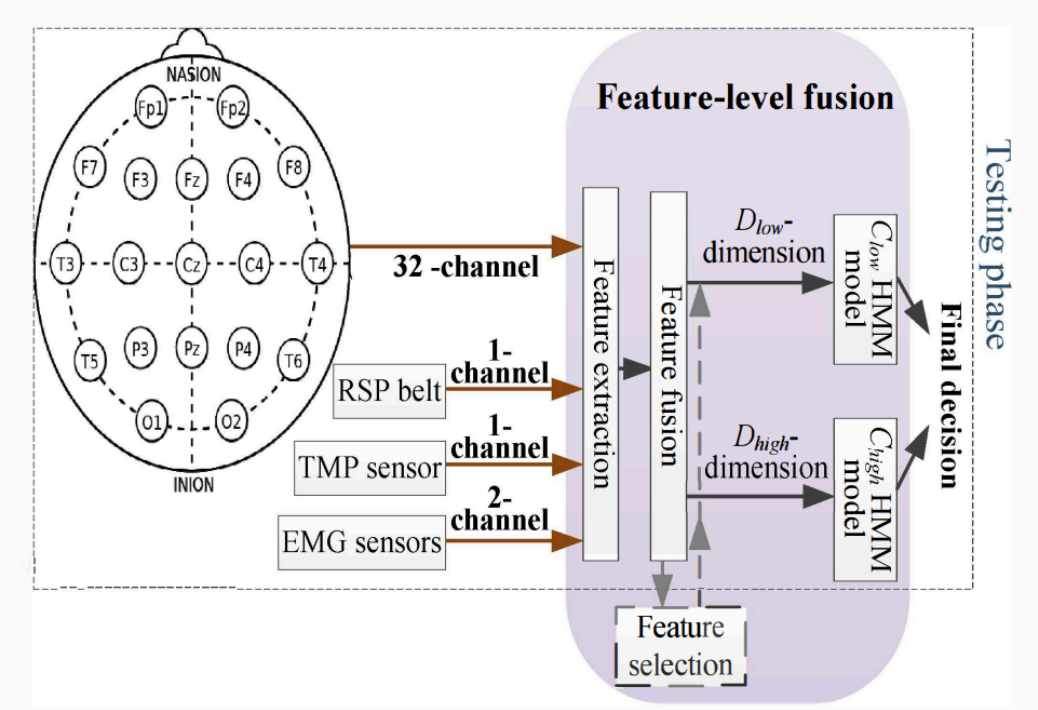


Figure 5.10 - Feature-level Fusion example for Emotion Recognition on the particular example of paper 26.

Since not all extracted features are relevant to emotion processing, a lower dimensional feature subset is identified from the concatenated feature space derived from four modalities to recognize the emotion based on their valence and arousal level.

This method is an example of how could be implemented a fusion of multiple biomarkers CNN model. In this example and in our project, the emotion classification must be performed on all studied biomarkers on the same emotion space : here it is the valence-arousal emotion space. One significant challenge is to already find a suited dataset for eye tracking regarding a valence-arousal emotion space, as well as the analysis for the GSR data that was recorded in the experiment but not statistically studied yet. An other challenge is to map the eye tracking data that we obtained using the desktop application to an eye tracking dataset that uses arousal and valence as an emotion space, and the postprocessing software that calculates the position of the eyes on the website at any moment of the experiment finally consists in a starting point towards emotion recognition when we experiment on a website.

Conclusion

The desktop application presented in this project could function regarding the data gathering on 20 participants that could be made. Using postprocessing compensated the fact that scrolling could not be recorded on the website experiment. Besides, a first glance at the eye tracking data could be provided with the heatmaps and scan paths that were generated for each participant.

One challenge yet to overcome regarding the desktop application is to add the face emotion recognition algorithm that is working in real time without facing all the lags that came with it.

Regarding postprocessing, a future work regarding this project could be to implement a CNN based on an emotion space based on valence-arousal so that a fusion-level feature is performed on an eye tracking classifier, the actual face emotion recognition classifier, and possibly a GSR based classifier.

Bibliography

[1] Blascheck, T., Kurzhals, K., Raschke, M., Burch, M., Weiskopf, D., & Ertl, T. (2014, June). State-of-the-Art of Visualization for Eye Tracking Data. In *EuroVis (STARs)*. http://www.profitippliga.de/papers/58.pdf

[2] Russell, J. A. (1980). A circumplex model of affect. Journal of Personality and Social Psychology, 39(6), 1161–1178. https://doi.org/10.1037/h0077714

[3] Lim, J.Z.; Mountstephens, J.; Teo, J. Emotion Recognition Using Eye-Tracking: Taxonomy, Review and Current Challenges. Sensors 2020, 20, 2384. https://doi.org/10.3390/s20082384

[4] "TobiiTech," [Online]. Available at: https://tech.tobii.com/technology/what-is-eye-tracking/

[5] Ming Jiang, Sunday M. Francis, Diksha Srishyla, Christine Conelea, Qi Zhao, Suma Jacob, Classifying Individuals with ASD Through Facial Emotion Recognition and Eye-Tracking : https://www-users.cse.umn.edu/~qzhao/publications/pdf/embc2019.pdf

[6] C. Meng and X. Zhao, "Webcam-Based Eye Movement Analysis Using CNN," in IEEE Access, vol. 5, pp. 19581-19587, 2017, doi: 10.1109/ACCESS.2017.2754299.

[7] EDPS TechDispatch on Facial Emotion Recognition
ISBN 978-92-9242-472-5 QT-AD-21-001-EN-N doi:10.2804/014217

https://edps.europa.eu/system/files/2021-05/21-05-26_techdispatch-facial-emotion-recognition_ref_en.pdf

[8] GSR eSense Online Manual, available at : https://mindfield-esense.com/esense-skin-response/index.php

[9] What is Python? available at : https://www.python.org/doc/essays/blurb/

[10] Anaconda (Python distribution), available at : https://en.wikipedia.org/wiki/Anaconda_(Python_distribution)

[11] Python interface to Tcl/Tk, available at : https://docs.python.org/3/library/tkinter.html

[12] Peirce, J., Gray, J.R., Simpson, S. et al. PsychoPy2: Experiments in behaviour made easy. Behav Res 51, 195–203 (2019). https://doi.org/10.3758/s13428-018-01193-y

[13] Niehorster, D.C., Andersson, R. & Nyström, M. Titta: A toolbox for creating PsychToolbox and Psychopy experiments with Tobii eye trackers. Behav Res 52, 1970–1979 (2020). https://doi.org/10.3758/s13428-020-01358-8

[14] CefPython3 documentation, available at : https://github.com/cztomczak/cefpython

[15] OpenCV Library, [Online]. Available at : https://opencv.org/about/

[16] Viola, Paul & Jones, Michael. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. IEEE Conf Comput Vis Pattern Recognit. 1. I-511. 10.1109/CVPR.2001.990517.

[17] Selenium overview, available at : https://www.selenium.dev/documentation/overview/

[18] About pandas, available at : https://pandas.pydata.org/about/index.html

[19] Seaborn: statistical data visualization, available at : https://seaborn.pydata.org/

[20] Khan, M. Q., & Lee, S. (2019). Gaze and Eye Tracking: Techniques and Applications in ADAS. *Sensors (Basel, Switzerland)*, *19*(24), 5540. https://doi.org/10.3390/s19245540

[21] Ghose, U., Srinivasan, A.A., Boyce, W.P. *et al*. PyTrack: An end-to-end analysis toolkit for eye tracking. *Behav Res* 52, 2588–2603 (2020). https://doi.org/10.3758/s13428-020-01392-6

[22] A. Mollahosseini, B. Hasani and M. H. Mahoor, "AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild," in *IEEE Transactions on Affective Computing*, vol. 10, no. 1, pp. 18-31, 1 Jan.-March 2019, doi: 10.1109/TAFFC.2017.2740923.

[23] Zheng, Wei-Long & Dong, Bo-Nan & Lu, Bao-Liang. (2014). Multimodal emotion recognition using EEG and eye tracking data. 2014 36th Annual

International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC 2014. 2014. 5040-3. 10.1109/EMBC.2014.6944757.

[24] F. Noroozi, M. Marjanovic, A. Njegus, S. Escalera and G. Anbarjafari, "Fusion of classifier predictions for audio-visual emotion recognition," *2016 23rd International Conference on Pattern Recognition (ICPR)*, 2016, pp. 61-66, doi: 10.1109/ICPR.2016.7899608.

[25] Jing Chen, Bin Hu, Lixin Xu, P. Moore and Yun Su, "Feature-level fusion of multimodal physiological signals for emotion recognition," *2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2015, pp. 395-399, doi: 10.1109/BIBM.2015.7359713.