UNIVERSITY OF CAMBRIDGE

MPhil in Machine Learning and Machine Intelligence 2021-2022

Module Coursework Feedback

Module Title: Designing Intelligent Interactive Systems

Module Code: MLMI10

Candidate Number: J902C

Coursework Number: 1

I confirm that this piece of work is my own unaided effort and adheres to the Department of Engineering's guidelines on plagiarism. √

Date Marked: Click here to enter a date. Marker's Name(s): Click here to enter text.

Marker's Comments:

This piece of work has been completed to the following standard *(Please circle as appropriate)*:

	Distinction			**Pass**			**Fail (C+ - marginal fail)**		
Overall assessment (circle grade)	**Outstanding**	**A+**	**A**	**A-**	**B+**	**B**	**C+**	**C**	**Unsatisfactory**
Guideline mark (%)	90-100	80-89	75-79	70-74	65-69	60-64	**55-59**	**50-54**	**0-49**
Penalties	**10% of mark for each day, or part day, late (Sunday excluded).**								

The assignment grades are given **for information only**; results are provisional and are subject to confirmation at the Final Examiners Meeting and by the Department of Engineering Degree Committee.

Eye-Gaming: shooting assistant design

J902C

1 PROBLEM CONTEXT

It is well known that the number of people playing video games, also known as gamers, has been constantly increasing in the last decades (fig. 1) because of several factors. First, the computing power of personal computers or other devices has increased drastically in the last decades, allowing the majority of the population to play video games in their own computers or mobile phones. Additionally, video games have become a core part of our culture and have an impact beyond the traditional video game community. Games are focusing on providing social experiences more than ever before, communities are stronger and create movements that span way beyond the actual game world. Connectivity in-home and on the go allows players to stay connected to their games and their friends regardless of time and location. Most games are not a one-time experience any longer but rather a service that connects people and keeps them entertained for many years.

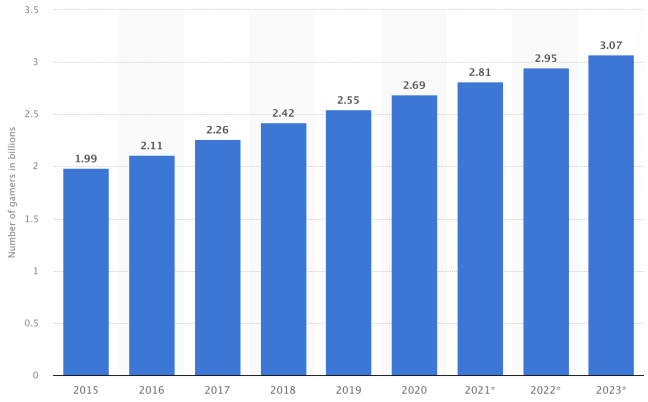

Figure 1: Number of gamers in billions, from 2015 to 2023 (expected), via Statista. (https://www.statista.com/statistics/748044/number-video-gamers-world/).

Unfortunately, there is a group of the population that cannot afford to play the vast majority of video games: handicapped or disabled people. Video games can make them feel happy and are a great distraction from the daily pain that they might be living with. Also, playing video games can boost creativity, improve problem-solving skills, and cultivate teamwork. People with disabilities, however, have limited opportunities to enjoy video games: those with visual challenges may not be able to participate in games that are rich in graphics, and those with mobility challenges may not be able to use a standard game controller. In particular, people with upper-body movement disorders struggle to play video games where they have to be very precise, for example, in shooting video games. Shooting video games tend to consist in some sort of military simulation whose goal is to defeat the enemy team by killing their members or conquering their base, using to do so weapons. Since the player has to aim to shoot properly their weapons, and move effectively to improve their positioning in the battle field, he or she should be as skilled and coordinated as possible.

1.1 Solution-Neutral Problem Statement

In order to alleviate this problem, the goal of this project is to create a system that allows disable people with upper-body movement or coordination disorders to play shooting video games.

Since there are a lot of types of disabilities and they might have different grades of severity, this project is focused on handicapped people that are not able to move or coordinate the muscles of their upper-body. Some examples of diseases that can cause the former are: spasticity [12], cerebral palsy [9], severe gait disorders [4] or severe Parkinson [10]. The software will help this group of people and it might will be useful for those with other similar disabilities.

In this document elaborates the design of the desired system, called *Eye-Gaming*, an assistant for shooting video games.

2 REQUIREMENTS SPECIFICATION

After defining the target audience and the goals of the project, the next stage is requirement elicitation. To facilitate to carry out this phase functional analysis is performed. Figure 2 illustrates a FAST diagram that describes the main function of the system.

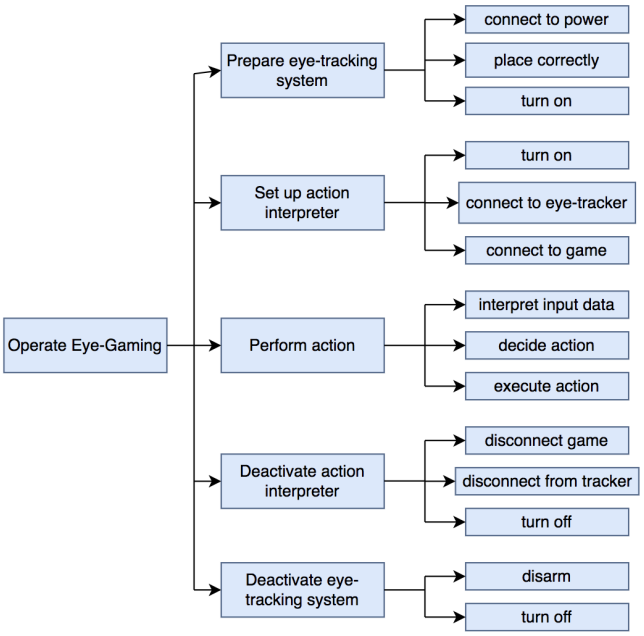

Figure 2: FAST Diagram: main needs to be accomplished

MLMI 10, Designing Intelligent Interactive Systems,

There are 3 basic needs to be accomplished: set up the eye-tracking system, configure the action interpreter and connect the interpreter to the game. Other functionalities can also be added, like the possibility of incorporating a voice command recognizer to give feedback to the system in case the user is able to talk. This would help the interpreter identify the desired action. In addition, all the systems can be disarmed from the computer and game.

To gather all the necessary requirements, it is important to follow some general guidelines to not miss any important point. To do so, a requirement checklist matrix is used and illustrated in figure 3. The main considered requirements are marked as green, and the ones that are not important for this project and that they have not been considered are marked as red.

		Process	Performance	Safety	Cost	Documentation
Product in Use	Operation	Operation Process	Operation Performance	Operation Safety	Operation Cost	Operation Documentation
	Maintenance	Maintenance Process	Maintenance Performance	Maintenance Safety	Maintenance Cost	Maintenance Documentation
	Disposal	Disposal Process	Disposal Performance	Disposal Safety	Disposal Cost	Disposal Documentation
Product Design/ Manufacture/ Supply	Design	Design Process	Design Performance	Design Safety	Design Cost	Design Documentation
	Manufacture	Manufacturing Process	Manufacturing Performance	Manufacturing Safety	Manufacturing Cost	Manufacturing Documentation
	Distribution	Distribution Process	Distribution Performance	Distribution Safety	Distribution Cost	Distribution Documentation
	Installation	Installation Process	Installation Performance	Installation Safety	Installation Cost	Installation Documentation

Figure 3: Requirement checklist matrix

In the following subsections the system is further detailed, listing the requirements specification.

2.1 Regulations and Standards

The product is thought to be commercialized in Europe and the US, so both territory regulations and standards must be accomplished. *Eye-Gaming* is not a medical device, but some elements are in close contact with skin and eyes. Additionally, the intended users are people with certain health issues, so some medical standards have to be followed. In the list below all the considered regulations and standards are listed.

- FDA Tripartite Biocompatibility Guidance for materials.
- BS 8444-3:1996 (IEC 60300-3-9:1995): Risk management - Guide to risk analysis of technological systems.
- BS EN 60601-1:1990 (BS 5724-1:1989): Medical electrical equipment and general requirements for safety.
- AAMI HE48:1993 : Human factors engineering guidelines and preferred practices for the design of medical devices.
- IEC 60204-1:2000 : Safety of machinery - Electrical equipment of machines - General requirements.
- QSR 820.30 - ISO 7.3.6: Design validation to ensure device meets user needs.
- QSR 820.170 - ISO 7.5.1: Installation, inspection and test instructions and ensure product is installed accordingly.

2.2 Functional requirements

(1) The eye-tracking system shall read eye movements and encode them numerically.
(2) Eye-tracking system shall communicate the interpreter the encoded eye movement in real time.
(3) Interpreter system shall translate eye movement data to a desired action.
(4) Interpreter shall communicate the game the action desired to execute.
(5) The system should have an alternative data source for the interpreter, such as a voice recogniser, that the user could use to clarify some actions.
(6) The system should have two modes: In ON mode the system is functioning and consuming energy; and in OFF mode it is shut down.

2.3 Physical requirements

(7) The eye-tracking system shall weigh less than a kilogram.
(8) The eye-tracking system should weigh less than 400 grams.
(9) The eye-tracking system shall be smaller than 40x20x20cm.
(10) The eye-tracking system shall perform effectively in a range less than 5 meters away from the screen of the game.
(11) The material of the eye-tracking system shall be safe in contact with skin and eyes.

2.4 Interface requirements

(12) The software system shall indicate when the eye-tracking system is correctly connected.
(13) The software system shall indicate when the action interpreter is correctly set up.
(14) The shooting assistant software should indicate what action is being performed in real time.

2.5 Performance requirements

2.5.1 *Adaptability requirements.*

(15) The system shall be compatible with all Call of Duty games previous 2022.
(16) The system shall be compatible with Fornite version 10.20 and former versions.
(17) The system shall be compatible with Counter Strike version 2021.

2.5.2 *Availability requirements.*

(18) The system should be operating 99.5% of the time.
(19) The system shall be operating 97% of the time.
(20) The system should identify and perform the correct action 90% of the time.
(21) The system shall identify and perform the correct action 75% of the time.
(22) The average number of system downfalls shall be less than 3 every 100 game plays.
(23) The mean time between system failures shall be less than an hour.

2.5.3 *Reliability requirements.*

(24) The mean time to system failure shall be less than 20 hours.

(25) The system shall be initialised again in less than 3 minutes in case of failure.
(26) The system shall be able to detect 1 action per second.
(27) The system should be able to detect 2 actions per second.
(28) The system should have an additional commander system, controlled by voice, where the user can repeat or specify the desired action.

2.6 Environmental requirements

(29) The hardware should be set up in an empty room, where the game is played.
(30) There shall not be any physical obstacle between the user, the eye-tracking system and the game screen.
(31) The system shall be used under convenient temperature conditions for the user, between 10 and 25 Celsius degrees.

2.7 Maintenance requirements

(32) The eye-tracking system shall be calibrated every time it is used.
(33) The eye-tracking system should be cleaned every time it is used.
(34) The eye-tracking system should be stored in the provided container/bag when it is not being used.
(35) In case the eye-tracking system is damaged, the average time to repair should be less than 3 working days.
(36) The interpreter software should be updated when a new version is released.

2.8 Disposal requirements

(37) The hardware part of the system shall be environmentally friendly disposable.
(38) The hardware part of the system shall be disposed in a computer recycling centre.

2.9 Validation requirements

(39) The interpreter shall be tested and simulated before deployment.
(40) The interpreter shall have at least 75% performing accuracy when decoding actions.
(41) The eye-tracking system should operate in real time, providing data every second.

2.10 Manufacturability requirements

(42) The software should be directly available to purchase and download.
(43) The production of the hardware part should be greater than 100 items per week.

2.11 Distribution and storage requirements

(44) The hardware part of the system shall be stored in the main warehouse until departure.
(45) The hardware part of the system should be delivered in less than 5 working days to Europe, and less than 10 working days to America.

2.12 Installation requirements

(46) The system shall operate in Windows 7 or Windows 10 versions.
(47) The system should operate in MacOS High Sierra 10.10 or newer.

2.13 Cost requirements

(48) The hardware part of the system shall cost less than £100 to produce.
(49) The distribution of the hardware system shall be less than £5 per item.
(50) The customer base price shall be greater than £500, but smaller than £2000.
(51) The customer price for the consumed energy shall be smaller than £1 per hour of game play.

2.14 Documentation requirements

(51) The system shall be provided with a manual to set up the eye-tracker, the interpreter and to connect to the desired game.
(52) A free helpline shall be available to answer customer questions about installation and set up.
(53) Risk analysis document shall be developed.
(54) Validation and verification documents are required.

3 CONCEPTUAL DESIGN

After defining the requirements, the goal now is to arrive at a single conceptual design that links functions to function carriers or solutions.

First, the overall function of the system and its boundary are described in figure 4. The diagram illustrates how *Eye-Gaming* uses an eye-tracker to translate eye movements into executable actions in the shooting game. In order to facilitate this problem, *Eye-Gaming* also uses a voice recognizer (like any type of microphone) to allow the user to give verbal commands to help the interpreter infer the desired action.

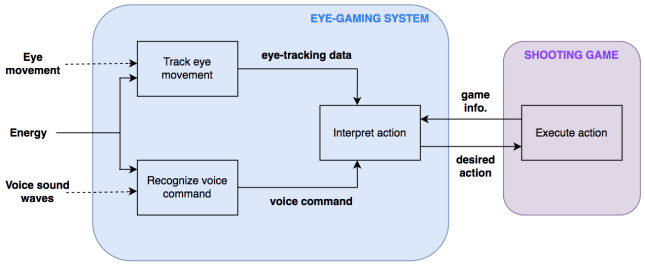

Figure 4: Functional diagram of the overall functionality of *Eye-Gaming*

As it can be seen, the eye-tracker device receives the actual eye movement that has been read with some kind of sensors (detailed in the following sections) and transforms all this information into interpretable data. Similarly, if a voice recognizer is employed, it is able to transform voice sound waves into numerical data. The

interpreter uses the interpretable generated data and the environment information provided by the game to infer what is the desired action, that is sent to the game to be executed.

The key function of the entire system is "interpret action", since it is here where the majority of data processing is done and where the action is determined. Figure 5 shows the decomposition of this function, modelling its main inputs and outputs, as well as the uncontrolled parameters.

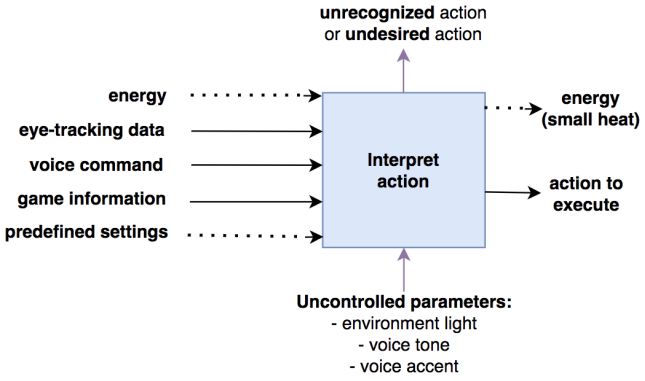

Figure 5: Modelling of "Interpret action" function

The individual component whose mission is to interpret the desired action receives the decoded eye movement data, information about the current state of the game and potentially some voice command. With all this information, the module shall generate an action to execute, based on the predefined parameters such as inference settings, expected time between two actions, game configuration, etc. consuming energy to do so, and emitting some resulting energy in form of small heat. The uncontrolled parameters are transferred from other functions, regarding difficulties to capture eyes' actions and voice.

Once the overall functionality of *Eye-Gaming* is defined and how it is decomposed into simpler functions, the conceptual design has to be created by establishing a mapping between all functions and function carrier or solutions. The table shown in figure 6 collects all the considered functions of diagram 4 and proposes several solutions for each function. First, the system has be fed with energy. It can be obtained connecting to electrical sockets or using different types of batteries (single use batteries or rechargeable). The eye-tracking system can infer where the user is looking at the screen by tracking rotations of the eye typically using one or a combination of four methods: projecting infrared light towards the participants' eyes (Purkinje images [6]), detecting users' gaze via a regular camera [3], using electrodes that are placed above/below or left/right of the eye (Electrooculography [13]), or employing contact lenses with some form of sensor [8]. On the other hand, the user voice could be recorded using different types of microphones (regular or condenser) or reading user's lips. To infer the action, a ML inference method could be used, as well as a lookup table that maps eye movements to actions. Finally, the predicted action could be executed using a mechanical system to handle the game controller, or directly using software, such as a driver to communicate the interpreter with the game, or calling a game API if it is available.

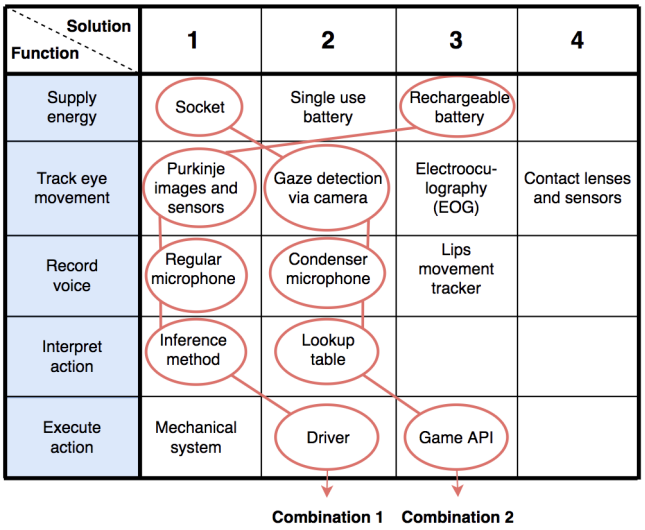

Figure 6: functions and function carriers matrix, known as morphological chart, with considered combinations

After collecting several solutions for the functions that models the system, some suitable combinations are selected. In table 6 we can see two considered combinations. The first of them considers a rechargeable battery for the system, the usage of Purkinje images and sensors to track the eye movement, a regular microphone to record the voice commands, ML inference methods to compute the action and a driver to communicate the action to the game. The second one would get the energy from sockets, use a camera for gaze recognition, a condenser microphone for voice commands, a lookup table to interpret the desired action and use the game API to execute it.

After generating some combinations, the next step is to identify the trade-offs and exploring the design space of the combinations to obtain the best concept for the final system.

Table shown in figure 7 illustrates the concept weighting and comparison. The considered criteria takes into account several factors that are important for the final product, such as eye-tracker size and quality, voice recognizer weight and quality, system delay, prediction accuracy or total cost of the product. Each factor receives a weighting representing its importance and it is evaluated taking an integer value between 1 and 5 if it is a positive factor, i.e., the bigger the value the better (e.g. predicted action accuracy); or an integer value between -1 and -5 if we are interested in minimizing the criteria (e.g. total cost).

For each criteria of table 7 a parameter analysis has been executed. By analysing the parameters we make sure to evaluate correctly and objectively both concepts and get an unbiased final concept recommendation. Figure 8 includes the analysis of two parameters: detected action accuracy and total cost of the system. Prior checking this illustration, remember that the detected action accuracy measures the proportion of correctly classified actions

CRITERIA	Weighting	CONCEPT 1		CONCEPT 2		IDEAL
		Value	Wt. x Val.	Value	Wt. x Val.	Wt. x Val.
Tracker weight	3	-1	-3	-2	-6	-3
Tracker size	4	-2	-8	-2	-8	-4
Tracker appearance	1	3	3	4	4	5
Tracker quality	4	4	16	2	8	20
Voice recognizer weight	3	-1	-3	-2	-6	-3
Voice recognizer size	3	-1	-3	-2	-6	-3
Voice recognizer appearance	1	2	2	3	3	5
Voice recognizer quality	3	2	6	4	12	15
System delay	5	-1	-5	-1	-5	-5
Predicted action accuracy	5	4	20	3	15	25
Total cost	4	-2	-8	-2	-8	-4
			17		3	48

Figure 7: Concept weighting and comparison

(the taken action was the desired one), and the total cost refers to the customer price (device + software + installation).

The estimation of the accuracy value is not linear with respect the actual accuracy value. For instance, a 30% accuracy is almost useless. That's why a 70% accuracy is evaluated with 3, and a 90% or greater is translated to a value of 5. A tolerable value is 3 (70% accuracy), since this is the threshold where the final system would be considered useful. On the other hand, the customer price can be linearly evaluated. In this case, negative integers are used because the bigger the cost the worse since we are looking for an affordable final product.

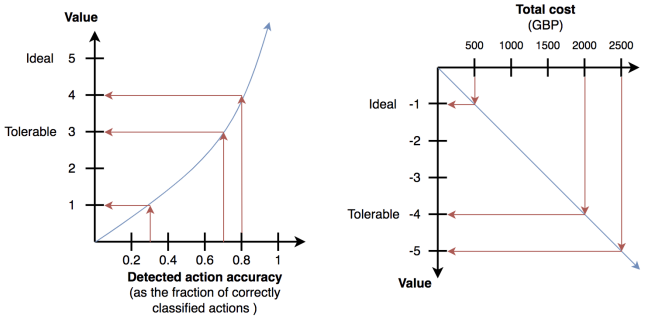

Figure 8: Parameter analysis: action selection accuracy (left) and customer price (right)

Analyzing table 7, the two concepts considered in the morphological chart 6 can be compared. The first combination is expected to have a lighter eye-tracker than the second one, since a system to capture Purkinje images is usually smaller than a regular camera. However, a camera could look better in the gaming room, compared with the advanced device needed to get Purkinje images, that's why the eye-tracker appearance of concept 2 is greater than the first concept. On the other hand, the eye-tracker based on Purkinje images is known to be quite more precise than a regular camera to capture gaze, this is the reason why concept 1 outperforms concept 2 under this criteria. Focusing now on the voice recognizer, a condenser microphone is clearly heavier, larger and it has considerably better quality than a regular microphone. Concept 2 is slightly better under this criteria.

Both concepts are considered to have almost no delay, that's to say that the latency between the actual human movement and the execution of the predicted action is almost negligible. An inference method would be quite fast in test time, and a lookup table to generate decisions would also be really rapid. However, the ML inference method can map a huge amount of eye movements to game actions using little memory space, where a look up table would need to be enormous to handle all the possible movements and, in some cases, the movement would not be stored in the table and the desired action could not be identified. Finally, both concepts are estimated to have approximately the same customer price.

Note that concept 1 would execute the predicted action using a software driver. Concept 2 would use instead a game API, so it would be dependent of the game company to actually publish and maintain an API to communicate to the game the desired actions.

Overall, after the weighting and comparison of table 7, concept 1 is preferable. It has got an overall evaluation of 17, greater than the evaluation of concept 2, that is 3. The recommended system after this design phase would use a rechargeable battery, Purkinje images and sensors for eye-movement detection, a regular microphone for voice recognition, a ML-based inference method and a built-in driver to communicate the system with the shooting game.

4 EMBODIMENT DESIGN

In this section the conceptual design is taken into a prototype level. First, the general architecture of the system is sketched in figure 9 describing the overall workflow. The user is wearing the special glasses equipped with *Eye-Gaming* system. The product can read eyes' movements using Purkinje images and some sensors (1). Additionally, the glasses have a microphone that the user can optionally employ to give more accurate commands to the gaming software. All the gathered data from the glasses are wireless transferred (2) to the shooting assistant app to interpret it and execute an action (3). The data emission between the glasses and the computer can be done by Wi-Fi or Bluetooth.

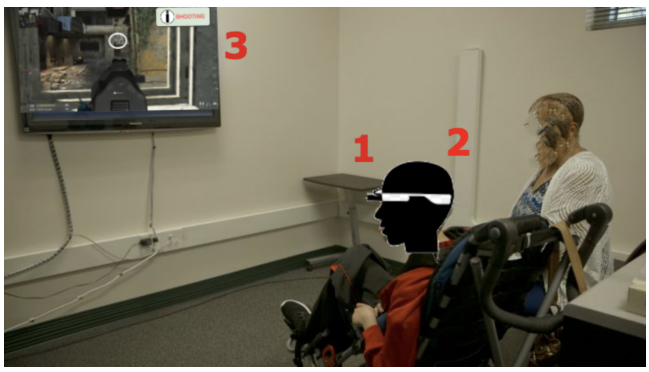

Figure 9: Sketch of the general architecture of the system

The principal components can be further described. The main hardware component is clearly the glasses that incorporates the eye-tracking system and the microphone. A realistic mock-up is included in figure 10. The glasses include an eye-tracker (1) that projects a light into the user's eye to generate Purkinje images and read his/her gaze. In one of the sides, a microphone (2) is incorporated in order to capture any potential voice command. Additionally, the glasses are built with sensors (3) near the eyes to help measuring their movement. Finally, a signal transmitter is able to send all the data to the computer app to be further processed. The glasses are mainly made of titanium, that is a durable and skin-safe material. Flexible parts like the frame are made of titanium plastics, and the lenses and the eye-tracker emitting laser are made of glass.

Figure 10: Eye-tracker and microphone mock-up

In the computer where the game is played, *Eye-Gaming* app can be installed. The user can install and configure the drivers (software to communicate with the game), check that the eye-tracker is successfully calibrated and connected and modify its settings. Moreover, in the app the user can select what game he/she is playing and set up the shooting assistant. All of this is represented in figure 11.

After considering all the necessary hardware, an estimated bill of the materials can be drafted:

- Glasses lenses: £80
- Glasses frame: £25
- Eye-tracker laser: £120
- Eye-tracker sensors: £75
- Microphone: £10
- Assembling: £5
- TOTAL PRODUCTION COST: £315

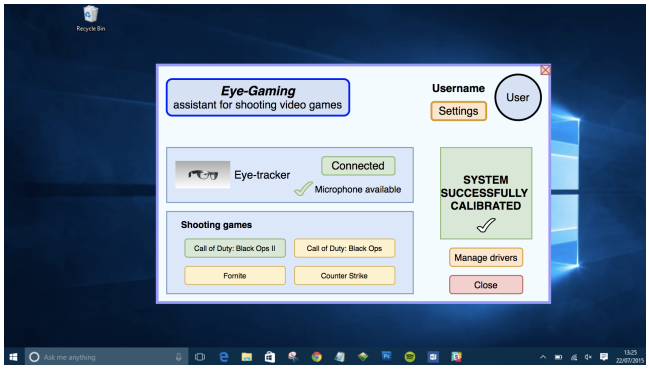

Figure 11: Sketch of the software platform of the system

To sum up, the overall workflow of the system is illustrated in the diagram of the figure 12. The system can be correctly configured following the workflow, installing the app and the drivers, calibrating the eye-tracker after placing it correctly and link the glasses to the computer. Once everything has been configured, the game can start and the eye-tracker will be reading the user's gaze and movements. The transmitter will send all this data to the computer, where a machine learning model (detailed below) will infer the desired action and communicate it to the game.

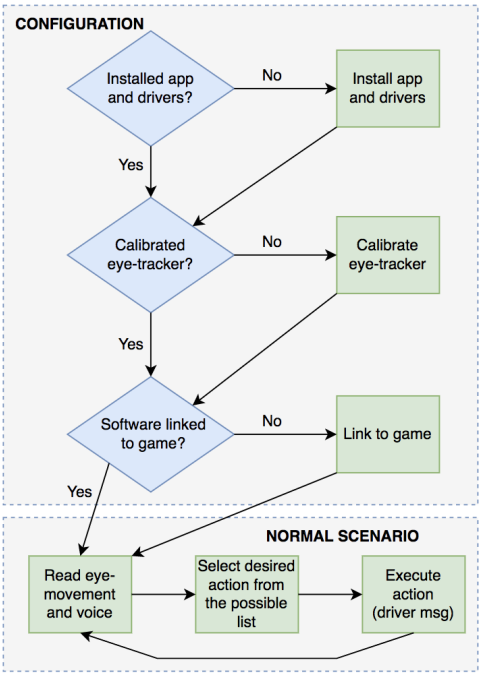

Figure 12: Overall workflow for carrying out the main functions

The most important feature of the system is the machine learning model that would infer the desired action given some input data. The considered model architecture is illustrated graphically in figure 13. The input data consists in the current game environment

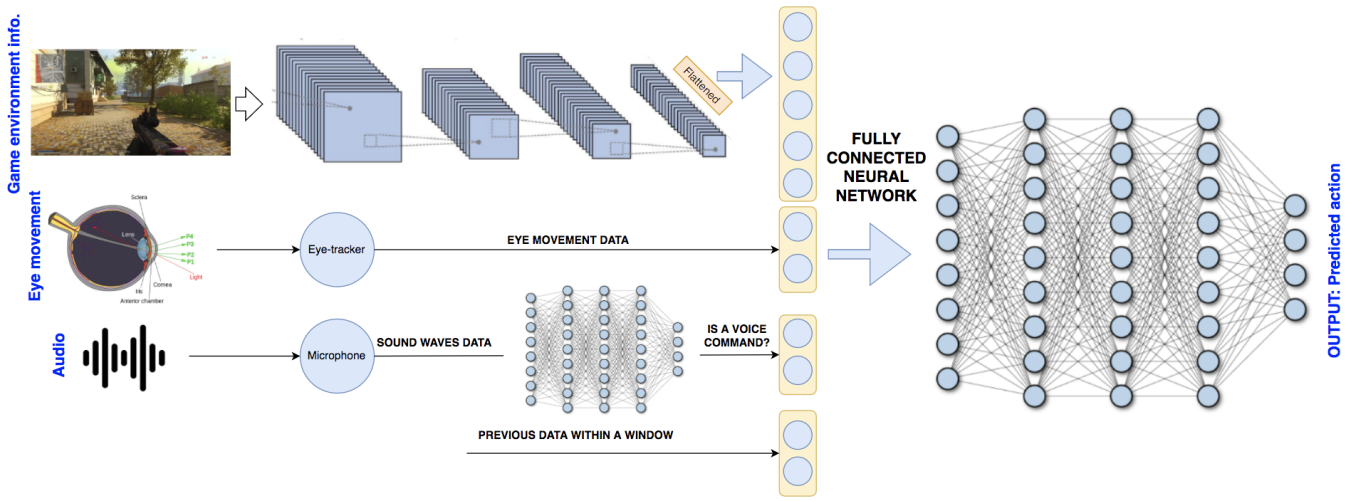

Figure 13: Machine Learning graphical model

(a) Walking **(b) Reloading** **(c) Shooting** **(d) Jumping**

Figure 14: Examples of actions. Images extracted from the game Call of Duty [1]

(described as a game frame), the user's gaze and eyes movements read by sensors, the voice commands (if available) and a set of previous analyzed data, such as the last game frames in a window and the last actions. Game frames are fed into a convolutional neural network [7], that would extract their main features and flatten them to be introduced to a fully connected neural network. The collected eye-tracker data, audio encoding and the previous analyzed features are also flatten and introduced as inputs for the fully connected neural network, that would need to be optimized to get as output the predicted action. Note that game frames are previously modelled by a convolutional neural network since it is good at capturing visual information. Moreover, eye movement data and voice commands can be modelled by different algorithms before feeding the FC neural network, such as Recurrent Neural Networks [5], that are good at capturing time-dependent and sequential features, or another FC neural network that would determine if the recorded sound is actually an intended voice command.

The majority of shooting games have a brief list of possible actions. Every game has its own unique features and actions, but in general all shooting games include and can be effectively played using just the following actions: walking, shooting, reloading weapons, changing weapons, jumping and crouching. The machine learning model has the goal of inferring the user's desired action from the previous list given the mentioned input data. A softmax function [2] can be placed at the output of the FC neural network to calculate the most probable desired action.

The data to train this model can be obtained by simulating a lot of games and recording player's gaze, the game environment and the real executed action for each moment. The data can be preprocessed later to build a dataset.

Figure 14 includes several examples of how *Eye-Gaming* performs in-game. The white circle indicates where the user is looking in real time, and in the right upper corner the software informs what action is predicted and executed. In general, the model would be able to infer that if the user is looking to the ground the desired action is walking in that direction (figure 14a); if he/she is watching to the weapon magazine, the desired action should be reloading it (figure 14b). Also, if the user is watching at other player, the desired action is highly probable to be shooting (fig. 14c). Similarly, if the user is walking towards a wall and he/she is looking at the top of it, the next action is probably jumping over it (fig. 14d).

5 RISK ASSESSMENT

Risk assessment is the analysis and evaluation of intended usage or purpose of the system. In this section we assessed the identification of hazards (potentials sources of harm) and their risks, how likely they are to cause a problem and how severe it would be if they did.

Table 1: Risk assessment of *Eye-Gaming* project using Structured What-If Technique (SWIFT)

Risk no.	What-if question	Causes	Hazards	Risk ranking	Action notes
R1	What if the user is able to talk and he/she is communicating while playing?	The user talks	The microphone captures the audio and this confuses the interpreter considering the user is giving voice commands	Low	Incorporate a button to deactivate voice commands
R2	What if the glasses battery runs out?	User forgets to charge the battery	The glasses disconnect to the software app and the system cannot determine an action	Low-Medium	Charge the battery
R3	What if the eye-tracker lenses and sensors are not calibrated?	User hasn't calibrated the system	The Purkinje image reader is not working properly and the desired action cannot be determined	Low-Medium	Calibrate the system
R4	What if the interpreter fails to correctly classify the desired action?	ML model prediction error	The performed action in the game is not the desired by the user, potentially losing it	Low-Medium	The ML model is expected to have at least 70% accuracy. To reduce this risk, the ML model should be further optimized
R5	What if the app suddenly losses the connection with the video game?	Connection error with the game server	The app is not able to communicate with the game and specify executed actions	Medium	Check the game server is up. If it is, try to restart the app and link to the game again
R6	What if the intelligent glasses lose the connection with the computer?	Wi-Fi or Bluetooth connection drop or transmitter failure	The glasses are not able to communicate with the computer app and specify the desired actions	Medium	Check the Wi-Fi or Bluetooth are working and set up. Check the transmitter is working correctly. Check the glasses are in connection range
R7	What if the electronic lenses that capture Purkinje images fail and damage user's eyes?	Overheating or malfunctioning	The user's eyes are damaged, causing vision problems or even blindness	High-Medium	The hardware, specially lenses, should be periodically checked

The chosen risk assessment procedure for this project is the Structured What-If Technique (SWIFT) that is a structured team-based study that uses "what-if" questions to help a team reflect and identify hazards and risks. It focuses on deviations from standard operations and the impact they may have on the system. I general, a What-if analysis consists of structured brainstorming to determine what can go wrong in a given scenario; then judge the likelihood and consequences that things will go wrong.

SWIFT is the chosen risk assessment procedure for several reasons. First, it is easy to use and there is no need for the usage of specialized tools. Additionally, since *Eye-Gaming* is focused for disabled people, SWIFT is really helpful in case we want the users to participate in a part of the risk assessment and they have no hazard analysis experience. Finally, SWIFT leads to deeper insight than other procedures, especially for person/people conducting the analysis.

Table 1 collects the SWIFT risk assessment developed for this project. All risks are enumerated and described, providing answers and clarifications using the What-If question. Risk R1 takes into account that some users might be able to talk, and the interpreter should be able to ignore the utterances that are not meant to be a helper command. Risk R2 contemplates that the user can forget to charge the battery. Similarly, R3 would be happening if the user forgets to calibrate the system. R4 considers the possibility that the ML model predicts erroneously the desired action. Moreover, risks R5 and R6 contemplate the scenario where any device lose connection, such as the glasses with the computer, or the app with the game. Finally, risk R7 is the most catastrophic among all of them, and evaluates the chance of damaging the user's eyes if the glasses break, overheat or malfunction.

Every considered risk has been assigned a score, based on its severity and likelihood of occurring. The scoring system is defined by a risk matrix, that allows individuals to rank the consequence and likelihood of an undesired event. The developed risk matrix is shown in figure 15, where all the given risks are placed accordingly their likelihood and impact.

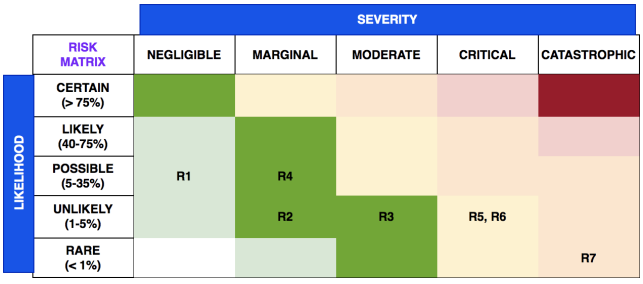

Figure 15: Risk matrix

In general, cells in the upper right of the matrix are not desirable, while those in the lower are likely to be acceptable. In our case, just R7 can be considered catastrophic, but since it is quite rare to happen it is not an unacceptable risk.

Table 2: Verification Cross-Reference Matrix for *Eye-Gaming*

Req. type	Requirement	Verification method	Success criteria
Functional	Interpreter system shall translate eye movement data to a desired action	Test	The interpreter is provided with an input example of decoded eye movement (example data of Purkinje images and sensors) and the output should be one of the considered actions
Physical	The eye-tracking system shall weigh less than a kilogram	Analysis	Weight the eye-tracker (electronic glasses) and check that it weighs less than a kilogram
Interface	The shooting assistant software should indicate what action is being performed in real time	Inspection	Assert that the software is displaying the executed action while playing, as shown in mock up 14
Validation	The interpreter shall have at least 75% performing accuracy when decoding actions	Test/Analysis	The data set to train the ML model is divided in 5 folds and an estimation of the performing accuracy is obtained using one fold as test set (technique known as 5-fold cross-validation). Assert that the accuracy is higher than 75%
Environmental	There shall not be any physical obstacle between the user, the eye-tracking system and the game screen	Inspection	The technician or the user that operates with the hardware can visually check that there are no physical obstacles
Performance-Adaptability	System shall be compatible with Counter Strike v.2021	Demonstration	Demonstrate that the system is capable to perform using Counter Strike shooting video game

6 VERIFICATION AND VALIDATION

In the **verification** phase we make sure that we have built the system right. The requirements listed and described in the requirements specification in section 2 are evaluated to ensure that the system complies with them.

The first employed verification technique is the usage of a verification cross-reference matrix (VCRM) that specifies in detail how each requirement is verified. Table 2 collects some examples of how some requirements have to be verified, including req. type, requirement statement, verification method and success criteria. The verification method explicitly specifies how the requirement shall be tested (visually, demonstration through system manipulation, checking some inputs return expected outputs, etc.).

Other verification phases include code correctness. In our case, we make sure our software works correctly. This can be done by executing unit tests (automatic tests that compare actual system outputs with expected outputs), code reviews (engineers review the code for correctness and compliance) and dynamic program analysis (analyzing the code while the system is working, to identify potential bugs or performance bottle-necks).

Finally, the user interface has to be examined. This can be done by also using unit tests, or even using expert human testing: an expert tries the system asserting requirements are satisfied. A non-expert human can also participate in the product verification. Every component is tested by a normal user against a set of guidelines, such as Nielsen's heuristics.

In the **validation** phase we make sure that we have built the right system. We assert that the solution-neutral problem statement of section 1 is assessed and this is the correct problem to solve. In the case of *Eye-Gaming*, the ultimate validation would be the market success, but before that other validation techniques can be used to check if the product can potentially fail or succeed.

Additionally, the machine learning model is examined based on sensitivity and validity. With sensitivity analysis we make sure the model is responding different outputs (actions) when we modify the inputs, i.e., when the user's eyes move differently. In model validation the generalization ability of the model is tested, making sure that the model is capable of correctly classify unseen movements or slightly different movements.

Heuristic evaluation is widely practiced, selecting a suitable set of heuristics. In this case, 10 Nielsen's heuristics [11] for usability inspection would be used to examine all elements of the interface and judging it:

(1) Visibility of system status: asserting that the software displays the glasses status, the interpreter status and whether the drivers are set up or not.
(2) Match between system and the real world: making sure that the system informs the user in natural language, not in a technical language.
(3) User control and freedom: asserting that the user can rectify if an undesired action is performed.
(4) Consistency and standards: following platform and industry conventions for shooting video games, using the corresponding jargon.
(5) Error prevention: asserting that the user is capable of calibrating and initiating the system without unintended actions.
(6) Recognition rather than recall: minimizing the user's memory load by making elements, actions, and options visible.
(7) Flexibility and efficiency of use: asserting that the user is capable of interacting with the game efficiently and smoothly.
(8) Aesthetic and minimalist design: making sure that the interface does not contain information which is irrelevant or rarely needed.
(9) Help users recognize and recover from errors: error messages should be expressed in plain language and suggest a solution.

(10) Help and documentation: assert that the product is well documented and help users understand how to complete their tasks.

Finally, *in-situ* evaluations are performed to check if a sample of the potential users are capable of employing the system as it is expected. Some expert users (engineers or developers that do not need to have disabilities but they know how the system works) and a sample from the target audience are asked to perform several actions to find if they are able to achieve the expected task, such as calibrating the eye-tracker, connecting the glasses to the software, connecting the app to the game, executing some actions using the system, etc.

7 SUMMARY

This document describes the design process of *Eye-Gaming*, a product with the goal of assisting people with upper-body movement disorders to play shooting video games. The user will be able to control and execute the main actions of a shooting game (shooting, walking, reloading, changing weapon, jumping and crouching) with just the movement of the eyes and, optionally, with some voice commands.

The final product is thought to have both software and hardware components. After considering several solutions in the first stages of the design process, we came to the conclusion that the hardware will consist in a system to track user's eyes movement and record his/her voice. This is done by a special electronic glasses that are able to capture user's gaze using Purkinje images [6] and some sensors that measure the muscles near the eyes. Also, a microphone can be incorporated into the glasses to record user's voice. The electronic glasses will transmit all the gathered data to the computer where the game is being played to be further processed. Moreover, the software will be able to infer the desired action given the eyes' movement and communicate it to the game. *Eye-Gaming* software application will inform the user of potential errors, the executed action, and whether the system ready to use or not. The inference is done by a machine learning model, capable of analyzing both visual and audio data, as well as the current environment of the game.

It is planned that the total product will have a production cost of £315 and a retail cost between £500 and £2000, depending on season, market studies and potential discounts.

The project entails several risks. The majority of them contemplate the possibility of a misuse from the user, such as not charging the battery or not calibrating the eye-tracker. All of them have a low risk because or the likelihood of happening is negligible or the impact is acceptable. The highest ranked risk is the scenario where the glasses malfunction and they damage user's eyes, that's why several inspections and quality controls have to be carried out regularly.

Finally, after executing the description of the product described in this document, it shall be tested using several methods. The code correctness is verified using both unit tests and dynamic program analysis. The whole system is validated before production using 10 Nielsen's heuristics and it is installed *in-situ* by an expert, i.e., a documented user or a technician.

In the future, the product could be improved to support other types of disorders and refined to enhance the user experience in the game.

REFERENCES

[1] Activision. 2022. Call of Duty. https://www.callofduty.com/content/atvi/callofduty/hub/web/en/home.html.

[2] Jason Brownlee. 2018. Softmax Activation Function with Python. Machine Learning Mastery. https://machinelearningmastery.com/softmax-activation-function-with-python/.

[3] Jixu Chen and Qiang Ji. 2009. 3D Gaze Estimation with a Single Camera without IR Illumination. 1 – 4. https://doi.org/10.1109/ICPR.2008.4761343

[4] Cleveland Clinic. 2022. Gait Disorders. https://my.clevelandclinic.org/health/symptoms/21092-gait-disorders.

[5] Douglas Coimbra de Andrade, Sabato Leo, Martin Loesener Da Silva Viana, and Christoph Bernkopf. 2018. A neural attention model for speech command recognition. *Engineering Aplications of Artificial Intelligence* (2018). https://doi.org/10.48550/arXiv.1808.08929

[6] Ji Woo Lee, Chul Woo Cho, Kwang Yong Shin, Eui Chul Lee, and Kang Ryoung Park. 2012. 3D gaze tracking method using Purkinje images on eye optical model and pupil. *Optics and Lasers in Engineering* 50, 5 (2012), 736–751. https://doi.org/10.1016/j.optlaseng.2011.12.001

[7] Zijin Luo, Matthew Guzdial, and Mark Riedl. 2019. Making CNNs for Video Parsing Accessible: Event Extraction from DOTA2 Gameplay Video Using Transfer, Zero-Shot, and Network Pruning. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*. Association for Computing Machinery, New York, NY, USA, Article 16, 10 pages. https://doi.org/10.1145/3337722.3337755

[8] Loïc Massin, Fabrice Seguin, Vincent Nourrit, Emmanuel Daniel, Jean-Louis de Bougrenet de la Tocnaye, and Cyril Lahuec. 2021. Smart Contact Lens Applied to Gaze Tracking. *IEEE Sensors Journal* 21, 1 (2021), 455–463. https://doi.org/10.1109/JSEN.2020.3012710

[9] NHS. 2022. Cerebral palsy. https://www.nhs.uk/conditions/cerebral-palsy/.

[10] NHS. 2022. Parkinson. https://www.nhs.uk/conditions/parkinsons-disease/.

[11] Jakob Nielsen. 2020. 10 Usability Heuristics for User Interface Design. https://www.nngroup.com/articles/ten-usability-heuristics/.

[12] American Association of Neurological Surgeons. 2022. Spasticity: Causes, Symptoms and Treatments. https://www.aans.org/Patients/Neurosurgical-Conditions-and-Treatments/Spasticity.

[13] Radwa Reda, Manal Tantawi, Howida shedeed, and Mohamed F. Tolba. 2020. Analyzing Electrooculography (EOG) for Eye Movement Detection. In *The International Conference on Advanced Machine Learning Technologies and Applications (AMLTA2019)*, Aboul Ella Hassanien, Ahmad Taher Azar, Tarek Gaber, Roheet Bhatnagar, and Mohamed F. Tolba (Eds.). Springer International Publishing, 179–189.