INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Detecting player's divided attention state during gameplay

Marco Martins Rodrigues

Master in Computer Science

Supervisor:
PhD Pedro Figueiredo Santana, Assistant Professor
Iscte - University Institute of Lisbon

November, 2021

iscte

TECHNOLOGY
AND ARCHITECTURE

Department of Information Science and Technology

Detecting player’s divided attention state during gameplay

Marco Martins Rodrigues

Master in Computer Science

Supervisor:
PhD Pedro Figueiredo Santana, Assistant Professor
Iscte - University Institute of Lisbon

November, 2021

”Try not to become a man of success. Rather become a man of value.”

Albert Einstein

Acknowledgments

I feel very grateful for all the support and help that my parents, Paulo and Inês, gave me during the development of this dissertation. Also to my brother Luís, who spent several hours testing the game and making sure to find bugs. To my supervisor, Pedro Santana, who was always insightful and patient in every step of the way. To my family and friends, for the kind words of encouragement.

Resumo

Os jogos sérios para a saúde e ensino estão-se a tornar mais comuns. Estes jogos têm evoluído os seus métodos de diagnosticar, tratar, e ensinar, não havendo, ainda assim, muito trabalho anterior relativamente à deteção do estado de atenção do jogador. Esta dissertação apresenta o desenvolvimento e validação de um sistema para detetar quando o estado de atenção do jogador é atenção dividida. O sistema utiliza mecânicas gerais e eventos presentes em jogos para analisar o desempenho do jogador e posteriormente detetar se o jogador está a dividir a sua atenção com outra tarefa. O sistema é testado com um jogo desenvolvido para o contexto desta dissertação. Os resultados obtidos mostram que o sistema é capaz de usar as metricas de desempenho do jogo para identificar a divisão da atenção por parte do jogador. O sistema pode ajudar designers a melhorar os seus jogos, tornado-os mais responsivos, e também pode ser uma framework para o desenvolvimento de outros sistemas, específicamente, para a deteção do estado da atenção do jogador.

Palavras-chave: videojogos; atenção dividida; jogos afetivos; interação pessoa-computador.

Abstract

Serious games for health and teaching are becoming more common. These games have been evolving their methods of diagnosing, treating and teaching; however, not much previous work has been done relatively to the detection of the player's attention state. This dissertation presents the design and testing of a system for the detection of divided attention state in players. The design of the system makes use of general game mechanics and events to analyse the player's performance and, from that detect whether the player is dividing attention with another task. The proposed system was tested on a game designed and developed in the context of this dissertation. The obtained results show that the system is able to use performance features to identify the divided attention state of the player. This system is expected to help developers improve their games, making them more responsive, and also be a framework of development for other systems, specifically, for the detection of the attention state of the player.

Keywords: videogames; divided attention; affective games; human-computer interaction.

Contents

Acknowledgments iii

Resumo v

Abstract vii

List of Figures xiii

List of Tables xv

List of Acronyms xvii

Chapter 1. Introduction 1

1.1. Context 1

1.2. Motivation 2

1.3. Research Questions 3

1.4. Objectives 3

1.5. Methodology/Development 4

1.6. Structure 4

Chapter 2. Literature Review 5

2.1. Engagement 5

2.2. Direct and Indirect Feedback 6

2.3. Reaction Time 7

2.4. Attention 8

2.5. Divided Attention State 9

2.6. Discussion 13

Chapter 3. Proposed Divided Attention Detection System 15

3.1. System's Overview 16
3.2. Information Gathering Module 17
3.3. Feature Prediction Module 20
3.4. Final Prediction Module 25
3.5. System implementation methodology 27

Chapter 4. Game Design and Development 29
4.1. Game Design 29
4.2. Game's dynamic difficulty adjustment 33
4.3. Game technical implementation 34

Chapter 5. Experimental Protocol 37
5.1. System Integration 37
5.2. Secondary Tasks 41
5.3. Experimental Procedure 42
5.4. Briefing 44

Chapter 6. Evaluation 45
6.1. Prediction metrics 45
6.2. Sub-module: Version A - feature selection 48
6.3. Sub-module: Version A - Combining features 51
6.4. Sub-module: Version A - Discussion 53
6.5. Sub-module: version B - feature selection 54
6.6. Sub-module: Version B - Combining features 56
6.7. Sub-module: Version B - Discussion 59
6.8. Discussion 60

Chapter 7. Conclusions 61
7.1. Future Work 62

References 65

Appendix 69

Appendix A. Tables with the performance features from Set 1. 71

Appendix B. Tables with the performance features from Set 2. 75

List of Figures

1 Focused Attention State and Divided Attention State. 15

2 System's Overview. 17

3 Time Windows. 19

4 Reference value for direct performance features. 21

5 Reference value for inverse performance features. 22

6 Interaction of the information gathering module and the feature prediction module 23

7 Interaction between the feature prediction module and the final prediction module 26

8 Game's screenshot with text overlaid in yellow. 30

9 Different projectiles. 30

10 Game's screenshot - reflecting a projectile. 31

11 Game's screenshot - selecting barrier type. 31

12 Game's cycle. 32

13 Game object's interactions. 35

14 Game's screenshot with overlaid zones. 39

15 Algorithm's diagram for the pairing of in-game events and reaction events. 40

16 Figures that the players must identify. 42

17 Game's screenshot - Images Task. 42

18 Game's screenshot - Countdown. 43

List of Tables

1 AND and OR operations. 25

2 In-game events and reaction events correspondence 37

3 Protocol Rounds Distribution 43

4 Possible prediction outcomes. 47

5 Sub-module: Version A - reference value from Set 1 applied to Set 1. 49

6 Sub-module: Version A - reference value from Set 1 applied to Set 2. 50

7 Score combined with wrong color. 52

8 On-time reaction to the launch projectile event combined with wrong color. 52

9 On-time reaction to the launch projectile event combined with score. 53

10 Sub-module: Version B - reference value from Set 1 applied to Set 1. 55

11 Sub-module: Version B - reference value from Set 1 applied to Set 2. 57

12 Standard deviation of the reaction time to the launch projectile event combined with score. 58

13 Wrong color combined with score. 58

14 Wrong color combined with the standard deviation of the reaction time to the launch projectile event. 59

15 Features from gameplay from the first round of the Set 1. 71

16 Features from gameplay from the second round of the Set 1. 72

17 Features from gameplay from the third round of the Set 1. 73

18 Features from gameplay from the fourth round of the Set 1. 74

19 Features from gameplay from the first round of the Set 2. 75

20 Features from gameplay from the second round of the Set 2. 76

21 Features from gameplay from the third round of the Set 2. 77

22 Features from gameplay from the fourth round of the Set 2. 78

List of Acronyms

ADHD: Attention-Deficit/Hyperactivity Disorder. 7, 16

DRT: Direct Response Task. 9, 10

ITS: Intelligent Tutoring System. 11, 12

SuRT: Surrogate Reference Task. 10

UML: Unified Modeling Language. 35

CHAPTER 1

Introduction

1.1. Context

Game development is a competitive industry, with numerous games developed every year with big investments backing them up. The success of a game depends on several factors, and some of them are out of the reach of the developers, for example, the different personalities of the players [1]. Research shows that people with different personalities desire different game mechanics [2]. This means that if a person is inclined for in-game self development, but the main game's mechanic is exploration, just like Teles et al [3], the player may not be able to enjoy the game as much, since the player is unable to find the desired mechanics to improve on. However, someone else can be oriented for exploration and so they may enjoy that game as it allows the player to explore the world.

The art of affective gaming focuses on the players' feelings and emotions, exploring means to provide a better experience for each player [4]. Player modeling is defined as attempting to measure each individuals' interaction with a game, and doing so by analysing the players' behaviour, cognitive and affective interactions, [5].

The main concepts studied regarding games and their interactions with the players are engagement, flow, immersion, presence, enjoyment, and attention. Engagement is a state of mind that can be achieved by interacting with a video game [1]. The result of engaging with a video game can be different as sometimes people may enter distinct states of mind. Immersion is a state defined as the users feel themselves extracted from reality [4]. If the players feel immersed, they may also feel present. Presence is a state of mind that is defined as felling part of the game, in other words the subjects physically feel themselves in the game [4]. The players may also lose their sense of time, which is typical of the state of Flow. Flow is defined as a state of mind in which the subjects are engaged and perform at their highest level. This state can be achieved when there is a good balance between the challenge presented and the subject's skills [6]. Enjoyment is subjective as the players do or do not enjoy a game. Attention is a psychological resource that everyone must use to execute any task, in this case, play a game [7]. To induce these states of mind in players, game developers create immersive scenes and focus on different aspects of the game, for example the game difficulty or emotions. This research focuses on attention and a possible method to detect the divided attention state of the player during gameplay.

1.2. Motivation

The highly competitive industry of game development is constantly in the search for innovation, which can make a game stand out and attract more people. This innovation can target the gameplay directly, by improving graphics or other elements in the game, or can target the gameplay indirectly by adding functionality that can improve the overall experience, for example, whenever the player character is idle for a certain period of time the character moves or talks. Another possible functionality could be suggesting a small gaming break when the player gets tired. These functionalities can, however, become unwanted if these happen too often or in inopportune moments. If the game could somehow identify the correct moments to use the new functionalities, these would most likely improve the player's experience. The correct moments could be when the player gets distracted from the game, or in other words when the players start dividing their attention with another task. For games to know if a player is not in a focused attention state there are two main approaches, using direct feedback or indirect feedback. The gathered information is then used to make a final assessment of the player's attention state. Both methods present clear advantages and disadvantages over the other. The direct feedback makes use of sensors that can gather physiological feedback, which are very useful, for example, to infer the stress levels of a player [8]. On the other hand, these methods require the use of external hardware, which, alternately, can be quite distracting, uncomfortable and can sometimes be expensive. The indirect feedback delivers a more accessible and long-term procedure to collect data as these are independent of any specific hardware, however the collected data is harder to interpret and use, leading to the development of more complex models or systems [8].

The research developed in the context of this dissertation focuses on creating and testing a system that uses the player's performance in a game to attempt to predict when the player is in a divided attention state. The indirect feedback is more suitable for this purpose as it is intended for the system to be independent of specific hardware. The requirement of external hardware is a deciding factor, considering that most gaming platforms do not make use of that type of equipment. To test the system, a game was purposefully designed and developed in the context of this dissertation.

1.3. Research Questions

There are several questions that are intended to be answered with this study. The questions are related to the divided attention state in games, how it affects the game, and how it could be detected. These questions are:

- If the players are dividing their attention with another task will their in-game performance be affected significantly?
- Is it possible to use the performance of the player to determine if the player is splitting one's attention with another task?
- Is it possible to create a system capable of detecting when a player is dividing their attention with another task, using only indirect features that represent the in-game performance of the player?

1.4. Objectives

This dissertation's main objective is to create a system capable of detecting divided attention in games during gameplay. The system should not require the usage of sensors and should be versatile. This objective is decomposed in a set of sub-objectives, namely:

(1) To design the system's main logic and components, with emphasis on using general indirect features that may be present in several games.
(2) To design and develop a game that provides an enjoyable experience to the player and that uses some of the most common features present in other games, capable of benchmarking divided attention detection systems.
(3) To develop secondary tasks in order to induce the player in a divided attention state, allowing for the system to be tested.
(4) To implement the system or part of it into the game, test this system with several participants, and compare the results of the system with the real attention state of the player.
(5) To identify the use cases of such a system and which flaws or limitation it may have.

1.5. Methodology/Development

This dissertation's was developed with the following methodology:

(1) Problem recognition: The main problem that was identified was the absence of an indirect feedback system or model that allowed a game to know whether the player was focused on the game or not.

(2) Hypothesis: The theoretical logic behind the proposed system is the differences in performance that occur between a focused attention state and a divided attention state on the tasks at hand.

(3) Design and development: The development of the game, the secondary tasks and the implementation of part of the system into the game.

(4) Validation: The usage of the game and system with different individuals so as to validate if the system was successful at detecting divided attention state or not.

(5) Conclusion: The final appreciation of the system addressing the different strengths and weaknesses of the system.

1.6. Structure

This research is organized in different chapters, each chapter develops a different topic related to the project. Chapter 2 overviews related works and important concepts, including similar projects and concepts that inspired both the system and the protocol. Chapter 3 describes the proposed system in detail, focusing on all the modules that compose the system and addresses different use cases. Chapter 4 describes both the game development and design, explaining the final stage of the game. Chapter 5 presents the experimental protocol used in the system's validation, describing both the interaction with the testers and the overall process. Chapter 6 presents and discusses the results obtained from the experimental protocol. Finally Chapter 7 draws conclusions based on the results and presents some possibilities for improvement as future work.

CHAPTER 2

Literature Review

This chapter presents the related work that provides valuable information for this research. This information focuses mostly on defining concepts and emphasising the most important parts from the related work.

2.1. Engagement

Engagement in games has been vastly studied. A systematic review [9] shares a lot of this research and is a good starting point to understand both what engagement is and why it has been a relevant subject to study. In this review, engagement is defined as a state resulting from the interaction of a player with a game, and, that the game is more engaging if the interaction is enjoyable [9].

One important aspect that affects the player's engagement are the conditions the player has, for example the interfaces through which the game is perceived [4]. To demonstrate this concept, Van conducted some tests [4]. These tests compared various situations, on some the players had a larger screen size and different sound pressure than the other situations [4]. The results end up demonstrating that the player's engagement and enjoyment were different in each singular situation. Another feature that can also influence the player's enjoyment and immersion is the difficulty of the game [6][10]. Difficulty adaptation can be automated, since it also helps the players to be immersed [10]. When a game can create a good balance between game challenge and player skill, the player has an easier time entering the Flow state [6]. Flow is an enjoyable state of mind that makes the players perform at their best [6]. Flow not only requires a balance between challenge and skill, but the balance should be of high value, in other words the challenge should be slightly above the skill of the player [6]. If not, the player will feel apathic and will lose interest right away [6]. So, it is important that game developers present not only a balance between skill and challenge, but also an adequate level of challenge. Cowley et al [6], also presents Flow as a composition of eight dimensions, which when fulfilled the player can enter the state of Flow.

One other important factor that affects the player's engagement is the player's personality, an overview [1] on the player's most played games and their personality traits demonstrates that different players enjoy games in unique ways. Each player may have their own goal orientation, meaning that a player may enjoy repeating and perfecting tasks, while another player may enjoy progressing in the story [2]. This study shows a cognitive emotional view of gamification, that creates a relation between the game mechanics, dynamics and the emotion and cognition. Mullins and Sabherwal [2] claim that this view is a model that can enhance the way that games generate emotions and feelings on the players. Mullins and Sabherwal [2] also explain that a good example of how to induce an emotion on the player is to allow their actions to generate consequences which allows the players to feel pleasure or displeasure. As an example, there are games with a certain number of lives. If the players lose one then they may feel sad or disappointed, but when they gain one they feel better. These concepts demonstrate that creating a successful game is a complex task. Some developers try to find different approaches to improve their games resulting, for example, in affective games.

2.2. Direct and Indirect Feedback

Affective Gaming is becoming another interesting topic for game developers, as the emotions that the players feel are a significant reason why they enjoy games. Affective Gaming is a concept that has been developed over several years [9]. A part of affective gaming is focused on the ability of a game to react and adapt to the player's emotions [11]. To predict these emotions the software needs information, from both the player and the game. This information can be gathered through direct feedback or indirect feedback. Direct feedback is obtained when the user is being monitored by sensors [8]. These sensors focus on physiological reactions from the player [8]. To study the attention states some researchers use direct feedback, for example, by the use of electroencephalograms in studies like [12][13]. On the other hand, indirect feedback is obtained by the game itself. Some studies also make use of this feedback to detect and slow down degenerative diseases, for example [14][15]. There are advantages and disadvantages of each approach.

The direct feedback allows the gathering of information like the heart rate or sweat levels. To acquire these data, the players need to be monitored by sensors, for example a heart rate sensor or a galvanic skin sensor. These sensors are often uncomfortable, however some are unobtrusive like a thermal camera used in stress analysis [16], not every player has access or can afford them, and the physiological reactions could be different between individuals [4]. The data gathered directly from these sensors is directly related to the person's emotions. As an example, if a player's heart rate increases, then it could be an indicator that player is afraid, or, if the heart rate is low then it could indicate that the player is calm. There is a research that makes use of direct feedback to create a Physiological Response Prediction [17]. The Physiological Response Prediction framework learns how each player reacts physiologically to different in-game situations and eventually is able to predict these physiological reactions based on the game state [17].

The indirect feedback is the information extracted from the inputs of the players, or other methods that do not relate directly to the physiological state of the player. An example of a feature that can be extracted is the frequency of the inputs or the reaction time of the player. Some people may press the buttons faster because they are excited and others because they are angry. This means that it is not easy to identify the players emotions from these indirect features. More data is required to make a good prediction because the data does not have a direct relation with the emotions. As an example an eye tracker can be used to track the player's gaze and then adapt the gameplay accordingly [18]. The facial expressions of the players have also been used to alter gameplay [19]. The positive side of the indirect feedback approach is that it could be used by more people and without many hardware dependencies.

2.3. Reaction Time

The player's reaction time to events is a feedback that games can calculate and use as indirect feedback. The reaction time is affected by several factors and these have been studied extensively. Reaction time has been studied per individual over several sessions [20], it has also been studied with people affected by Attention-Deficit/Hyperactivity Disorder (ADHD) and other diseases, in order to diagnose or treat them [21][22][23]. The reaction time has also been studied by comparing different people with different habits, like the reaction time between athletes and non-athletes [24] or between eSports players and athletes [25]. The reaction times can also vary depending on the type of stimulus that the person has to react to, for instance, there is a clear difference in reaction time between a visual stimulus and an auditory stimulus, with the later being faster [26]. This can be a deciding factor for game developers when using reaction times to in-game events. The system proposed in this dissertation was tested with different performance features, including the reaction time to in-game events, since these can be calculated in the game and can be an indirect performance feature.

2.4. Attention

This section describes attention according to McDowd [7]. Attention is the intentional action of focusing and concentrating on a task, which can also be seen as a human resource. Attention is influenced by a lot of physiological factors, for example, stress and fatigue. Attention can also be split into several tasks, a good example would be cooking and having to answer a phone call at the same time. If the person doing the tasks does not exceed their limit of attention capacity, then the person can do both tasks without any problem. However, if the person's attention capacity is not enough for executing both tasks then either the tasks would be poorly performed, or one task would be done after the other.

Divided attention is the conscious act of splitting the attention over several tasks. The person must attempt to execute several tasks simultaneously. The example of cooking and answering the phone reflects this concept. But divided attention is not as simple as splitting the resource into several tasks. It was found that some difficult tasks done simultaneously were easier to perform than other simpler tasks done simultaneously. This observation originated the creation of models to attempt to identify the reason behind this phenomenon. One of these models is the Wicken's multiple resource model [27].

If two tasks would compete with the same resource dimension, then it would be difficult to perform the tasks simultaneously. However, if the tasks required a different resource dimension then they could be performed easily. The proposed model was composed of four resource dimensions, modality as the auditory and visual requirements, processing stage as the encoding and central processing requirements, information code as the spatial and verbal requirements, and finally the response mode as the vocal and manual requirements.

Attention switching is the usual alternative to the divided attention. The cooking example is also good to clarify attention switching. If the subject is cooking and needs to answer the phone, then the person could switch one's attention to the phone. This implies that the person is not paying attention to cooking. Attention switching also has costs in memory, so sometimes the divided attention could be a better option. The memory costs are the ability to remember the status of the task that was momentarily interrupted. For example, when the conversation on the phone ends, the person needs to remember the state of the cooking task.

Selective attention is the ability to choose one of many tasks and block the others. For instance, when a person is going on a train or metro and there is a lot of noise, the person can try to focus on a dialogue with someone. When a subject is unable to maintain selective attention on a task that means, it that the person got distracted. The selective attention is related to another concept entitled sustained attention. Sustained attention is the ability to sustain selective attention for a long period of time. The time factor introduces the concept of effort to keep the attention only on the chosen task. The sustained attention requires the capability to reject other events that may occur, for example, a thought or an external sound.

The performance differences between the focused attention state and the divided attention state of a person are the base of the system proposed in this dissertation.

2.5. Divided Attention State

The divided attention state was studied in relation to the cognitive workload, specifically in driving [28][29][30][31]. The driving was simulated for safety reasons. Driving is an interesting task to explore, as it is a complex activity that humans perform under divided attention [7]. To test the effects of divided attention on the cognitive workload while driving, Direct Response Tasks (DRT) were introduced. A DRT is a task that has a stimulus and the reaction time of the subject to the stimulus is recorded [28].

The following paragraph addresses Stojmenova and Sodnik's [28] research. A typical task is to ask the subject to press a button when a light is turned on. The various stimulus involved in the DRT may be a visual, auditory or tactile. These DRT are the metric used to compare the reaction times of the subjects while in the state of divided attention and selective attention. However it would not be possible to use DRT's in a selective attention state, since the DRT's are not part of the main task, in this case driving. To reduce the effect that DRT may have, the tasks are designed to be as less intrusive as possible. The main task is a composition of both simulated driving and the DRT, this main task simulates the selective attention state. The secondary tasks can be of two different types, some are visual-manual tasks and others are pure cognitive tasks. These are used in combination with the main task, driving and DRT's, in order to simulate a divided attention state.

The first type of secondary task is based on the visual attention being shared between a display and the environment. The example provided by the authors is defined as Surrogate Reference Task (SuRT), which consists on the subject identifying the area that contains the outlier in an image. The image is composed of several circles of the same size except one larger one. This task presents a good difficulty level variance, that depends on the difference of size between the outlier and the others, and on the size of the target area. Pure cognitive tasks are tasks that interfere as less as possible with the subject's senses and the answers from the subject are done vocally. The example is the delayed digit recall, which consists of listening to a certain amount of numbers, and telling them back out loud.

Stojmenova et al [29] presented a study that focuses on the use of the methods described to understand what type of stimulus is more sensitive for the cognitive workload, in other words, which type of stimulus affected more significantly the person's capability of executing the main and secondary tasks. The experiments consisted in three groups of participants. Each group would have a main task of simulated driving with a different DRT stimulus. For example, the DRT of the first group was based on visual stimulus, the second group auditory stimulus and finally the third group tactile stimulus. A secondary task was then introduced and consisted on the standardized delayed digit recall, in two different difficulties. Each group would perform this secondary task three times, each one with different input methods, first visual, then audio and then tactile. All the responses were performed vocally. The results demonstrated the visual response task had worse performance compared with the other two because driving requires a lot of visual attention. However, it was also found that all the stimuli were sensitive to the cognitive workload due to the differences in the response times. In this study [29], the DRT are used to monitor the user's workload, however this could be applied to games and serious games. As described, the DRT's consist in measuring the reaction times of the subjects in a specific task in order to evaluate the subject's performance at the task. These reaction could be measured by the game automatically, by creating events and measuring the time that the player takes to react to them.

One specific study proposed to monitor the player's attention on a serious game with the main purpose of fighting the Alzheimer's disease [15]. One factor that affects attention is the neurological status of the player [7], which means that if a person presents a neurological disease that person could present limited attention resources. As explained by Osman et al [15], Alzheimer's is most common in older adults, therefore the game had to be designed for seniors. The developers decided to use accessible equipment and did not implement difficult inputs. The playtests suggested that these could prove difficult for seniors to use. The game had to present an interesting context and an adaptive range of difficulties. The core concept of the gameplay was to take pictures of birds. The control system of the game consisted on a WiiMote-like controller and each picture could be classified with a different score, dependent on the bird's visibility. In this serious game [15], the intention was to detect when the player would get distracted and, in these moments incentivize the player to engage with the game again. A method to recapture the players' attention could be to surprise them like Itti and Baldi suggest, [32], since their study demonstrated that surprise methods are more reliable than other. Osman et al [15], explained that in order to detect if the player was distracted, the player's reaction times can be used. The current reaction time is compared to a mean value of the player's reaction time. The value of that difference could be a good indicator of the player's loss of attention. The tests done in the study were just related to reaction time and the variance of that same reaction time in different scenarios, more specifically in relation to the remote used. No tests were done with the Alzheimer's patients and it was not possible to find any further research or development in this project. In this dissertation, the proposed system is designed to be implemented with any game, allowing for the system to be employed for different use cases.

Attention is required to learn and, since Intelligent Tutoring System, (ITS), focus in digital methods to automate teaching, these can benefit of automated divided attention attention switching detection. ITS's have been studied for a long time and some researches have been addressing several maters to improve [33]. The following will cover two studies that address the identification of off-task behavior on students [34][35]. Off-task behavior is essentially when a student prioritizes any task other than the ITS tasks. This previous work was done on students over a long period of time. The purpose of having the data collection over a long period of time is to allow the students to naturally get bored of the ITS and present some off-task behavior. During the data collection process observers are present to monitor the student's behavior and classify it. These data varies from model to model, and is usually metrics that can be gathered by the system automatically, for example the number of errors and help requests. In 2007, Baker [35] presented a comparison between two models, the first model was focused on the usage of only time features, and the other model was focused on a total of 26 features. The most relevant was the speed variance on actions

[35]. Slow actions could be related to either off-task behavior or thinking on the problem, but if afterwards the interaction was fast, the model would perceive that the player was more likely to be on an off-task. In 2010, Cetintas et al [34] studied in a similar manner several personalised models based on different selected features. These features were time modeling; time and performance modeling; and finally, time, performance and mouse movement. Time modeling being the time in between actions, performance modeling, the correct answer ratio. The results showed that more complex the model is the better it performs. However, the authors also expanded and compared the models with personalization, that allowed the model to adapt to the player. This adaptation caused the results of the model to be more precise than the other models [34]. Cetintas et al [34] explains that the personalization is a method that is based on a general model and uses personalized data relative to certain tasks to then classify the behavior. The limitation presented is when only a small amount of data is present, which could cause the model to be less precise [34].

Some similar models were also studied in the serious game's context, more specifically in teaching games. The models were applied in two different studies each with a different game types. The methodology was the same as in the ITS's [36][37]. One of the studies [36] compares two different schools of different social status, to study the effect that the contact with technology may have in off-task behavior. Comparing the frequency of off-task behavior which was studied in different time periods. The results of this research showed that students that may have less contact with technology present less off-task behavior. Over the session of thirty minutes, the frequency of off-task behavior increased significantly in each ten-minute segment [36]. A collaborative game that only allows communication through an in-game chat also attempted to identify off-task behavior [37]. The messages in the chat are classified as off-task or on-task behavior based on information contained in the message, the context of messages and the timestamps of the messages. The accuracy of these models was 0.791 [37], showing that messages could also be a good indicator of off-task behavior.

These studies in serious games focused on the off-task behavior, which, as described, is the prioritization of tasks. This dissertation focuses on the divided attention, in other words, when the player performs two tasks at the same time.

2.6. Discussion

The literature review shows that even though divided attention has been studied in different situations, it has not been studied significantly in games, nor in serious games. To fill this gap in the literature, this dissertation proposes a system that exploits general indirect features to detect divided attention state. This enables the system to be applied to different games without requiring the use of sensors to monitor the player's state.

The differences in performance between the focused attention state and the divided attention state are what allows the herein proposed system to detect the player's attention state. This dissertation also proposes a system's deployment methodology to guide game developers, as well as some system's use cases for both videogames and serious games.

CHAPTER 3

Proposed Divided Attention Detection System

The proposed Divided Attention Detection System was designed to be versatile and to be independent of specific gameplay mechanics. The main purpose of the system is to be able to detect the time windows in which the players are in a divided attention state. The system makes use of the in-game performance of the player in different time windows, the performance being represented by indirect features extracted from the game, as an example, the score and reaction time of the player. The system is composed of different modules: the information gathering module; the feature prediction module; and the final prediction module. This chapter reviews the different modules that compose the system and the definition of time window in this context.

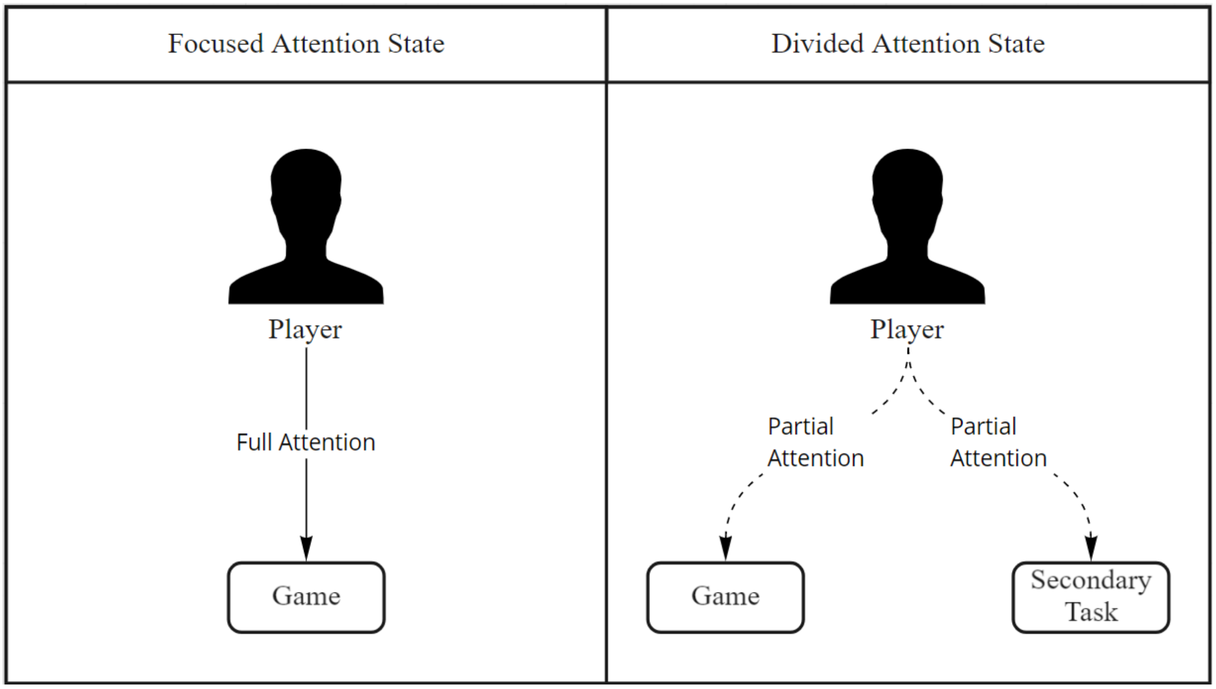

Figure 1. Focused Attention State and Divided Attention State.

The two attention states that the system is attempting to predict are the *focused attention state* and the *divided attention state*, depicted in Fig 1. The focused attention state is defined as a person being focused on the task at hand, which means that the person is actively performing the task and doing so the as best as possible. The divided attention state is defined as a person executing two tasks at the same time, which as discussed in Section 2.4, can hinder the performance at both those tasks. The system continuously gathers performance information, assuming that identifying a significant difference in the performance could be an indicator that the player's attention state changed. These expected differences in performance are what allow the system to predict the attention state of the player.

3.1. System's Overview

In a nut shell, the proposed divided attention detection system is intended to identify time windows in which the player is in a divided attention state. This system's approach, represented in Fig. 2, focuses on identifying the player's performance changes during gameplay. The system is implemented into the game, tracking the players' performance through the different features of the game. The feature tracking occurs during several time windows, which then enables the sub-modules of the system to compute a *reference value* per feature. Comparing the reference value with other feature values allows the system to create a prediction of the attention state of the player for each time window based on each feature. Afterwards, the system may use different features by combining their predictions, which is expected to improve the overall prediction of the system.

This proposed system was designed to be applied to most games, since the concepts used are common in games and in most cases easy to obtain. The system is flexible, enabling for the usage of different sub-modules, which allows for a more customizable system. The application of the proposed system depends on the game; for instance, once a divided attention state is detected the player can be surprised to catch their attention again, or suggest a gaming break, or give the player a clue on to how to proceed, or even adapt the game's difficulty. The system could also be used just for monitoring, allowing for an analysis of the parts of the game that the player was not in focused attention, maybe those parts were not interesting for that player. There are a large amount of possibilities for the usage of the system. Serious games for teaching could benefit from the usage of this system, since attention is a requirement for learning to occur. The system could also be implemented into serious games for health, for example, to help diagnose ADHD. The following sections detail each component of the proposed system.

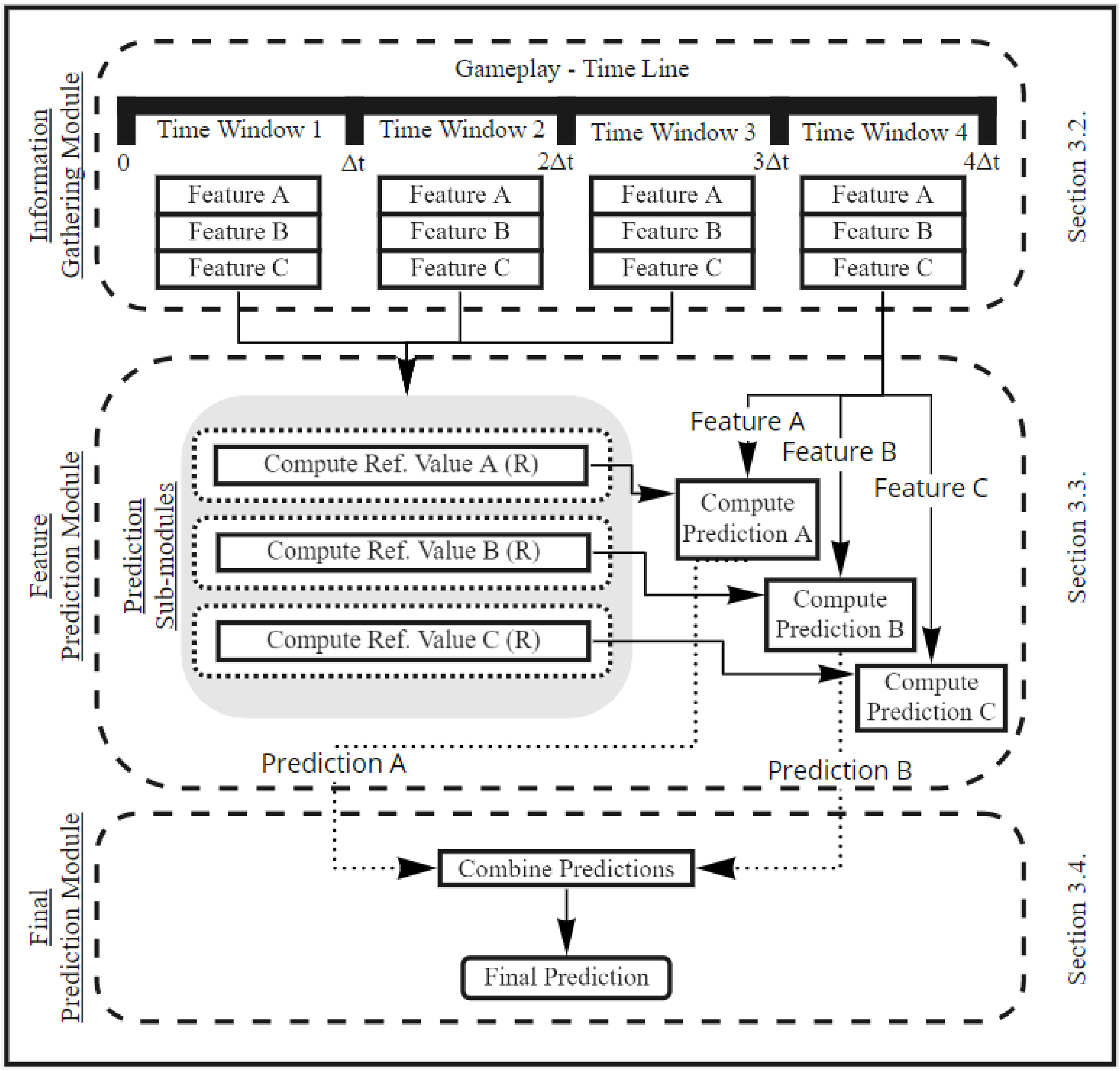

Figure 2. System's Overview - The information gathering module extracts feature values for each time window of gameplay. The feature values are then used in the prediction sub-modules to compute a reference value (R) for each feature, enabling the feature prediction module to make attention state predictions for any time window, one per feature. Lastly the final prediction module combines the previous computed predictions to make a final prediction.

3.2. Information Gathering Module

For the system to be versatile, it makes use of any game features that can be informative regarding the performance of the player. In general, games have several performance features, for example the score of the player, like in the game Tetris, or the player's health, like in Super Mario and so forth. These features are some of the most common in games, however, since certain games may focus on other mechanics and objectives, it may be more difficult to find performance features in those kinds of games, as an example, the Sokoban game. Since the objective is to organize a warehouse by placing boxes on the proper place strategically, the player may be mentally creating a strategy, without any input in the game. Without that input, there would not be any usable features.

In order to identify performance features, it is important to focus on the players' interactions with the game. Interactions differ from game to game and may not even be obvious that a certain interaction can be a performance metric. For instance, suppose a First Person Shooter, the main objective of the player is to hit targets. The obvious performance feature is the hit/miss shots ratio, whereas a not so obvious feature could be the number of shots per second. The player could be taking a long time to perfectly align each shot getting a perfect score, but if another person could also hit every shot in a shorter time span, then this person definitely showed a better performance. This example suggests that combining various performance features may improve the system.

To help recognize these features, some examples are presented: the reaction times to in-game events as the time that the player takes to react to an enemy attack could be an indirect performance feature. If the players are paying attention then it is more likely that their reaction time is faster than when their attention is split with another task. The speed at which an in-game task is completed can also be a performance feature, for example, the time the players take to solve a simple puzzle. Another example of a performance feature that may not be obvious is the number of times that players do not react to a certain game event, for instance, jumping over a trap. These and other similar features, present in games, could be tested and possibly used by the proposed system, however it is important to understand that different players may focus on different game play styles, leading to different features becoming better or worse for each player.

The proposed system has been devised to use features individually or combine several of them in order to make the final prediction of the player's attentions state. The expected advantages of combining several features are a more versatile system for different players, as the variance of the gameplay styles is less relevant due to the fact that the final prediction takes more information into account. On the other hand, doing so would possibly reduce the accuracy of the system to certain individuals. That is, if the predictions were very accurate for an individual, altering the combination of the features could reduce said accuracy.

The system operates over several time windows. A time window is a time interval in which the performance features are measured and associated to. Time windows may vary in duration (Δt) in between games, however must be constant within each game, as illustrated in Fig. 3. The duration varies between games due to several factors, one of these is the number of expected occurrences of an in-game event. If an event occurs frequently, then a smaller time window should be reasonable; however, other events may require larger time windows, for example, performance features that make use of reaction time to events. The duration may also vary, from game to game, considering that the system can only predict the state of the player for each time window. If a game can make use of several features, then the time windows for that game should be the minimum duration that allows all the features to be relevant. If the time windows are too short, then some features may not be useful, whereas if the duration is too large, then the predictions may not occur as often as desired. That is to say that since the system only predicts the attention state for each time window, the larger the time window is, the less predictions the system can make. Depending on the situation, it may be better to not use features that increase the duration of time windows, so as to increase the number of time windows that are predicted. For instance, if feature A requires a Δt of five minutes and all the other features require one minute, it may be better to not use feature A, allowing for the system to predict five times more attention states than if feature A was used.

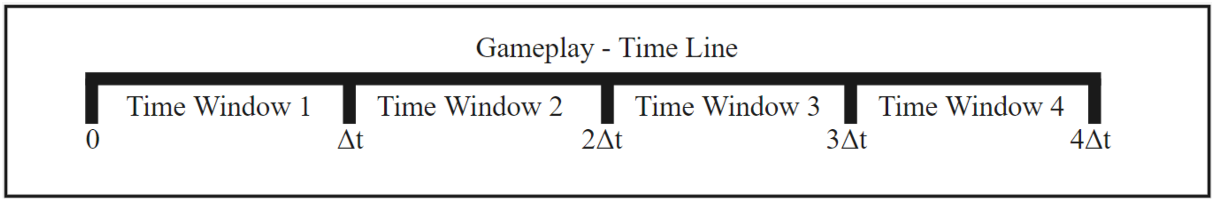

Figure 3. Time Windows.

The performance feature's value of the player is stored for each time window. As an example, the extracted value from the lives lost feature is stored in time window 1, then the new extracted value from the feature is stored in time window 2, an so on. The system gathers a large number of performance features across several time windows, for example, for half an hour of gameplay. These data is then used by the prediction sub-module, addressed in a following Subsection 3.3.1, to compute a reference value, which is then compared to the values extracted from the following time windows in order to predict the attention state of the player.

3.3. Feature Prediction Module

The in-game performance of the player is the key for the predictions of the proposed divided attention detection system. Some features have high values when the player's performance is high, like the score, and others have low values, like the number of lives lost. To clarify, the difference between these two features is how they measure the player's performance. The score is increased every time the player does something successful, which is directly connected to performance, a better performance implies a higher score. On the other hand, the lives lost increase every time the player loses a life, this is inversely connected to performance, a better performance implying a lower value of lost lives. Hereafter, features similar to the score are defined as *direct performance features* and features similar to the lost lives are defined as *inverse performance features*. This difference is important for the system, specifically for the sub-module, which behaves differently for these two types of features. The proposed system assumes that players should perform better when in a focused attention state and should perform worse when in a divided attention state.

The proposed divided attention detection system contains a sub-module that calculates a reference value. There is a reference value per feature and it allows the system to predict the attention state of the player, as depicted in Figures 4 and 5. The reference value separates the values of the feature that represent focused attention and divided attention.

3.3.1. Prediction sub-module

As mentioned, the prediction sub-module is responsible for calculating the reference value, depicted in Fig. 6. For this study, two alternative versions of the sub-module were developed and then tested. Both versions of the sub-module use a different method for each of the two types of features, i.e., the direct performance features and the inverse performance features. The distinction, between features, is done manually by the game designer before any calculations of the sub-module. Although the methods are different for the two types of features, the concept is the same for both. These methods were designed based upon the results of preliminary formative tests alongside trial and error experimentation.

The performance is represented by the values of the features that the game can extract. These feature values are organized by each time window, which therefore have a performance associated to. Given that the player's performance may vary even if the player maintains the same attention

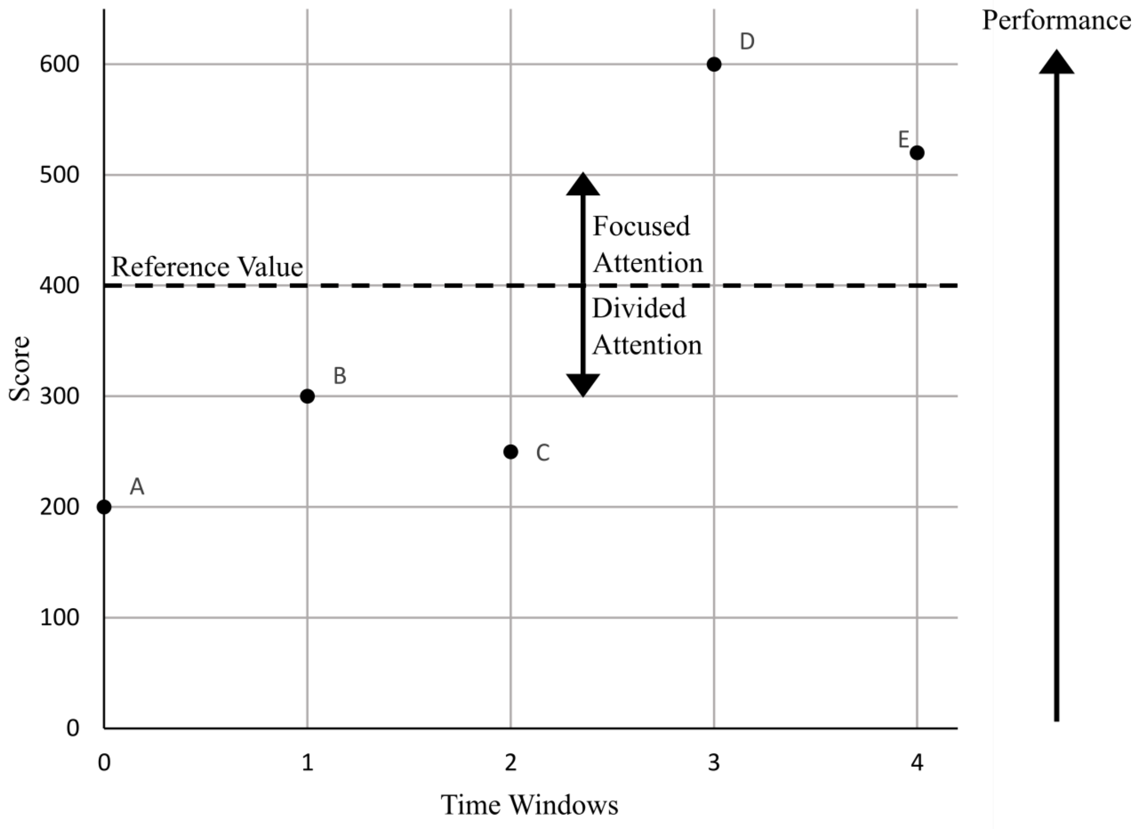

Figure 4. Reference value for direct performance features. Time windows 0, 1 and 2 present feature values below the reference value, thus predicted as divided attention state, whereas time windows 3 and 4 are predicted as focused attention state for being above the reference value.

state, the sub-module must take into account this variation when calculating the reference value. The two versions of the sub-module tackle this matter with different methods.

After a reference value is calculated, is then used to predict the attention state of the player in other time windows. For direct performance features, values above the reference value are predicted as focused attention state, since the performance is expected to increase when the feature values also increase, as depicted in Fig. 4. For inverse performance features, values above the reference value are predicted as divided attention state, given that the performance is expected to decrease when feature values increase, as depicted in Fig. 5.

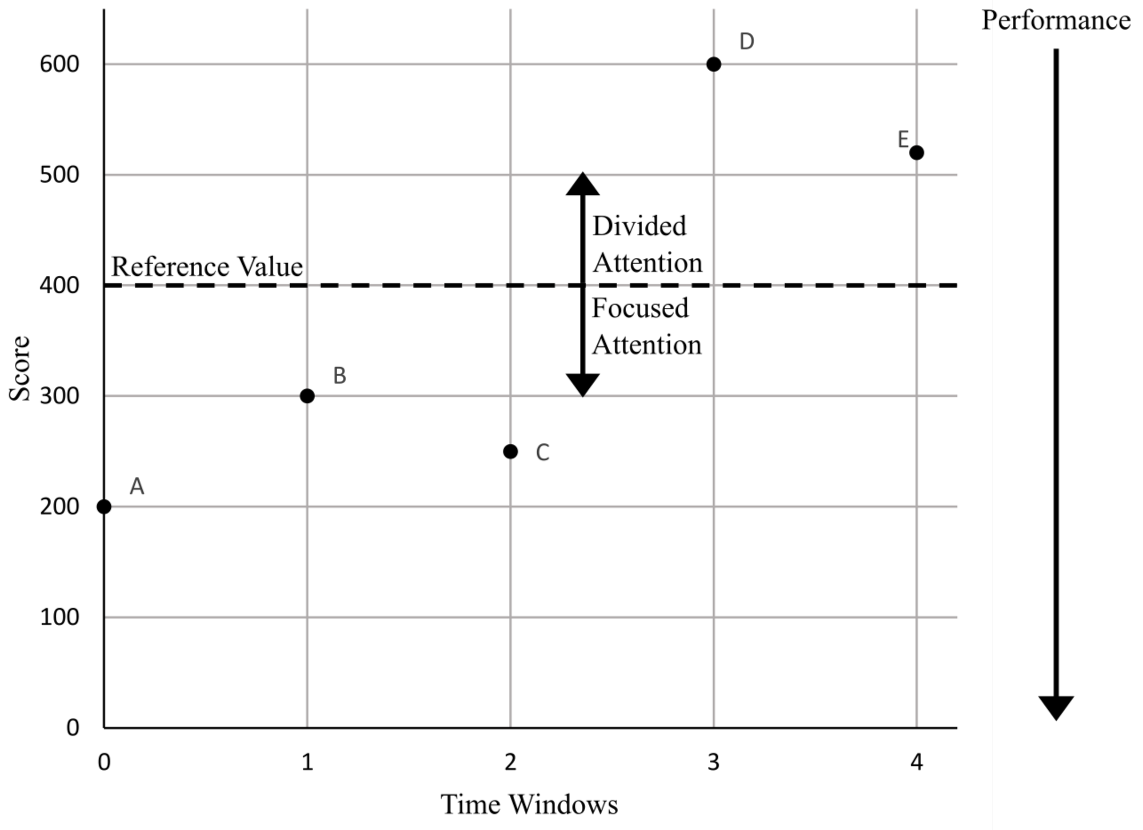

Figure 5. Reference value for inverse performance features. Time windows 0, 1 and 2 present feature values below the reference value, thus predicted as focused attention state, whereas time windows 3 and 4 are predicted as divided attention state for being above the reference value.

3.3.2. Prediction sub-module: Version A

This sub-module's version observes all the time windows to identify in which of those the player's performance is higher. This value could be used alone to compute a reference value, however, that reference value would not address the differences in performance that the players may have during gameplay. On the other hand, using the two highest performance feature values for two distinct time windows, p_1 and p_2, enables the reference value to be computed in a way that considers the players' performance variation.

For the direct performance features, the two selected time windows are the ones with the highest feature values, whereas for the inverse performance features, the two selected time windows are the ones with the lowest feature values. For instance, for the score feature, the selected time windows

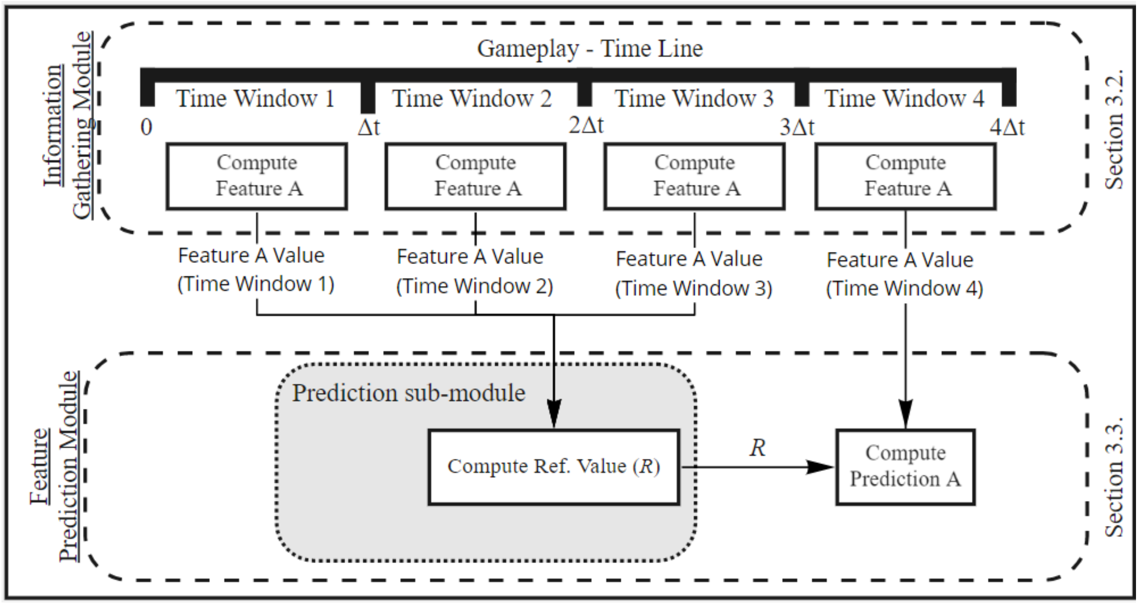

Figure 6. Interaction of the information gathering module and the feature prediction module. As an example, to predict the attention state of the player for a time window x, the feature values from time windows $x-1$, $x-2$ and $x-3$ are used in the sub-module to compute a reference value (R). The reference value is then used by the feature prediction module to make a prediction for time window x.

would be the ones with the highest scores; and for the lives lost feature, the selected time windows would be the ones with less lives lost.

The reference value for two given p_1 and p_2, $R_A(p_1, p_2)$, is computed depending on the type of feature, as follows:

$$R_A(p_1, p_2) = \begin{cases} \dfrac{p_1 + p_2}{2} - 2 \cdot |p_1 - p_2|, & \text{if direct performance feature,} \\ \dfrac{p_1 + p_2}{2} + 2 \cdot |p_1 - p_2|, & \text{if inverse performance feature.} \end{cases} \tag{3.1}$$

As mentioned, distinguishing the two types of performance features is crucial for the sub-module to function as intended. Since version A of the sub-module uses the two highest performance values, it means that, in all other time windows, the player's performance is lower. The reference value also needs to be lower than the highest values, otherwise, the system would only predict divided attention state. For this reason, the subtraction is used in direct performance features and the addition is used in inverse performance features. In this manner the system is able to predict both attention states according to the reference value.

There is a situation that can affect the predictions of the system, supposing that the maximum value of a feature is the same in two time windows; the reference value would be exactly the maximum value. In other words, the mean of two identical values is the same value, and the difference between them is zero implying that the reference value is the maximum value. In this situation the system would predict every time that the player in a divided attention state. This situation is unlikely to happen, however another version of the sub-module is presented.

3.3.3. Prediction sub-module: Version B

Version B of the sub-module selects the highest, p_H, and lowest, p_L, performance values of all time windows for each feature. The highest performance values are expected correspond to a time window played in a focused attention state, conversely, the lowest performance value is expected to correspond to a time window played in a divided attention state. Once again for direct performance features, the highest performance corresponds to the highest value and for the inverse performance features, the highest performance corresponds to the lowest value.

Unlike the version A of the prediction sub-module, which uses the two maximum values of performance, version B focuses on splitting the feature values according to the maximum and minimum values of the performance feature. This means that version B computes a reference value that is always in between the maximum and minimum values of performance, solving the possible issue that version A could create.

The reference value for two given p_H and p_L, $R_B(p_H, p_L)$, is obtained by subtracting or adding a third of the difference to the lowest performance value, p_L, depending on the type of feature, as follows:

$$R_B(p_H, p_L) = \begin{cases} p_L + \dfrac{|p_H - p_L|}{3}, & \text{if direct performance feature,} \\ p_L - \dfrac{|p_H - p_L|}{3}, & \text{if inverse performance feature.} \end{cases} \tag{3.2}$$

Once again, distinguishing the two types of features is essential for the sub-module to operate properly. Similarly to version A of the sub-module, version B also must adjust for the possible performance variation in focused attention state, hence the addition is used in the direct performance features and the subtraction in the inverse performance features, to avoid having the reference values exceed the maximum performance values.

Unlike version A of the sub-module, this version has a requirement, which is that the player must present at least one time window played in a divided attention state and another in a focused attention state. This requirement exists because the calculated reference value can only be an intermediate value between the two initial values, which implies that if the reference value is calculated only with focused attention state or only divided attention state time windows, the reference value would split the other time windows even though they were played in the same attention state.

This version could be used for a larger number of time windows, as the probability of the player being in both attention state is also greater. In other words, since this sub-module's version requires the player to present both attention states, it is better to be used with more time windows, which would increase the chance that the player present both attention states in those time windows.

3.4. Final Prediction Module

Although the system does not require the use of this module, it is expected that this module increases the system's ability to correctly predict the attention state of the player. The final prediction module uses the individual predictions computed by the feature prediction modules and combines the predictions from different features, depicted in Fig. 7. It is expected that the system benefits from combining features that are more reliable than to use all of them. In other words, the system is expected to increase its accuracy when combining features that, when used individually also have high accuracy values. As an example, if a feature A and a feature B both predict the attention state of the player correctly in most of the cases, combining their predictions could allow for better results because the features could complement each other.

Table 1. AND and OR operations.

Prediction		Operation	
A	B	AND	OR
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

In this research, the simplest method is employed to combine the predictions from different features, using the *And* and the *Or* binary operations, present in Table 1. Since these operations are used in binary, the predictions are converted to binary numbers. The focused attention state is converted to the value 0 and the divided attention state is converted to the value 1.

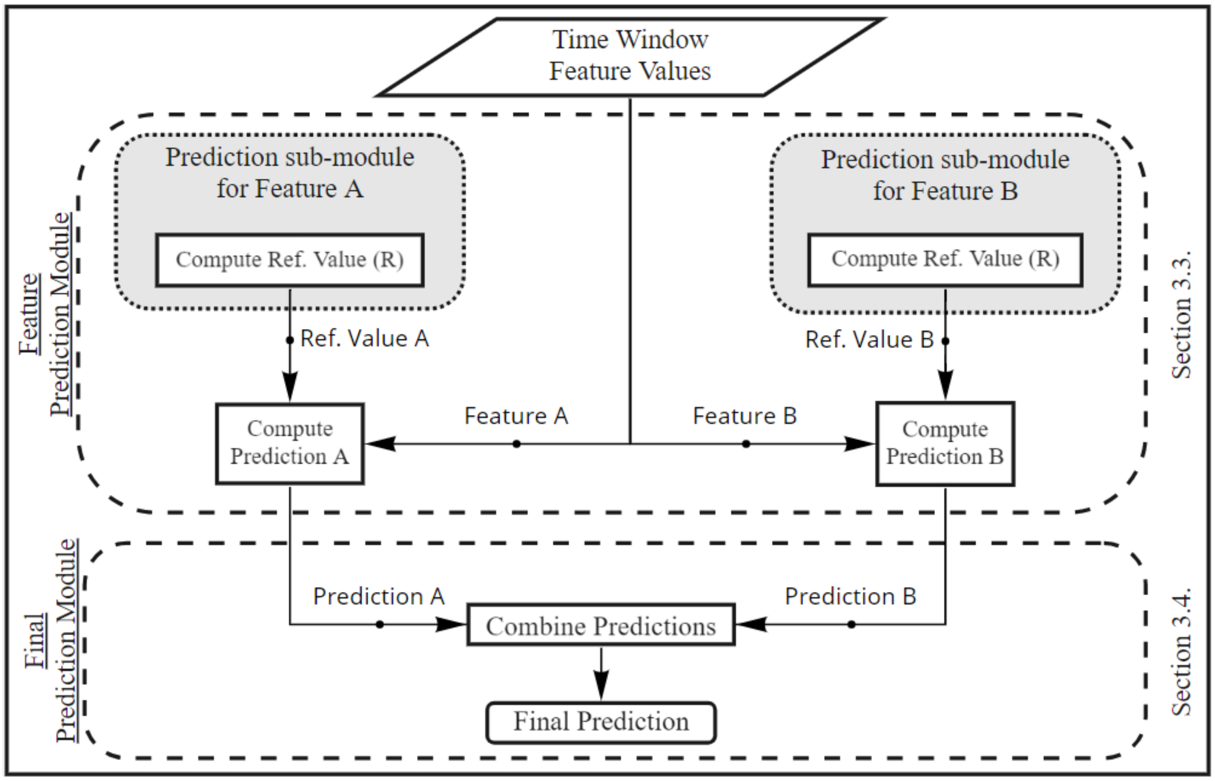

Figure 7. Interaction between the feature prediction module and the final prediction module. The predictions computed by the feature prediction module are combined by the final prediction module into a final prediction.

The final prediction can be used by the game in order to react according to the attention state of the player, provided such an adaptive scheme has been included in the game. The duration of the time windows is a factor that may influence the decision about which moment is the best for adapting the game. If small time windows are used, then the ideal moment is most likely after several predictions of divided attention state have been produced. For instance, the game could provide a clue to the player only if the system detects divided attention in three consecutive prediction moments, as it could indicate that the player is confused and not just performing a quick secondary task, such as answering someone's question. Conversely, if large time windows are used, waiting for consecutive divided attention predictions would lead to unfavourable delayed reactions, potentially causing the player to leave the game.

3.5. System implementation methodology

This section addresses in more detail the process of implementing this system. This methodology was developed to guide and help anyone who intends to use the system in their game. It is important to note that the system must be instantiated independently for each game segment (i.e., scene, level), since the performance of the players can only be tracked in each game context independently. In other words, each new reference values must be computed for each game segment. For instance, if in some parts of the game the player needs to shoot enemies, whereas in other parts to solve puzzles, the score of the player and/or the number of lives lost are most likely different, and so, correct attention predictions can only be produced if each situation is associated to its own reference values. There are three phases of implementation:

(1) System integration: The system is partially or totally integrated in the game. Allowing to predict the attention state of the player in different time windows.
(2) Feature selection: The system is tested to help the developers select the most relevant features and the methods of combining those features.
(3) System adaptation: The system is adapted so as to use the features and the combining method selected in the previous phase.

3.5.1. System integration

Starting with the information gathering module, the performance features present in the game must be identified, so that these can then be extracted and used by the system. The features must also be classified as direct performance features or indirect performance features. Afterwards it is important to analyse the frequency of the different features selecting the most appropriate time window duration.

To implement the feature prediction module, the sub-modules must be defined, for instance the version A of the prediction sub-module may be used for features A and B and the version B of the sub-module for features C and D. These process the different feature values to compute the correspondent reference values. Once this process is complete, the feature prediction module must be able to compute a prediction for the attention state of the player according to each individual feature. Finally, for the final prediction module to be integrated, the combining of features must be implemented. As mentioned, this process can be customized to suit each game differently.

3.5.2. Feature selection

After integrating the system, the game and system are tested, in order to identify which features predict the attention state of the players better. An example of a test that enables the identification of the best features is presented in Chapter 5. It focuses on inducing both attention states and use the system to predict each time window. An analysis of the results should provide enough information to select the best features. The best features depend on the game and how the predictions will affect that game. As an example, if the game's response to the divided attention state of the player is disruptive to the player in a focused attention state, then avoiding false positives should be the priority. This means that the best features are the ones that do not present any false positives.

3.5.3. System customization

In this phase the developers must adapt the system to predict the attention state of the player with the selected features and the selected combination methods. Once this process is complete, the system should be ready to be used. In other words, the system predicts the attention state of the player according to the selected features and the game can respond as intended. In the case that the developers are not confident in the predictions of the system the developers should repeat the previous phases.

CHAPTER 4

Game Design and Development

To test the divided attention detection system, a game was designed and developed. This game should provide an interesting challenge for the player, while at the same time provide different features to be used by the proposed divided attention detection system. The game should allow the player to learn how the game works and also how to improve on it. To explore the system's capabilities, the game must present several performance features in order to be able to compare their accuracy benchmarking. The description of the game as well as the relevant in-game mechanics are addressed in this chapter.

4.1. Game Design

The game occurs in a virtual room that has a heart in the centre (see Fig. 8). The players' main objective is to protect the heart from projectiles that are automatically fired against it, by raising two protection barriers at the right moment, which is done by pressing the up key on the keyboard.

There are projectiles stored in two containers, one on each side of the room by the ceiling. The projectiles differ in shape and color, as depicted in Fig. 9. At certain intervals, a random projectile is selected and fired by one of the two cannons, situated one at each side of the room. The firing intervals are slightly randomised to increase variability. The container enables the projectiles to fall into the cannon through a transparent pipe, which allows the player to identify which projectile type (shape and color) will be fired. Once the projectile is fired, its trajectory crosses over a protection barrier. There are two barriers, one on each side, shaped like a capsule. By raising the barrier, the player tries to intercept the trajectory of the projectile, blocking the path to the heart, Fig. 10.

Projectiles can collide with the protection barriers at different points and since these barriers are shaped like capsules, the projectiles can be reflected in different directions. Whenever a projectile is reflected to the ceiling, the player earns score points. This is an incentive for the player to improve the timing of the reaction in order to constantly send the projectiles up to ceiling and earn more score points. If the player is not precise in the timing, the barriers may miss the projectile and, therefore,

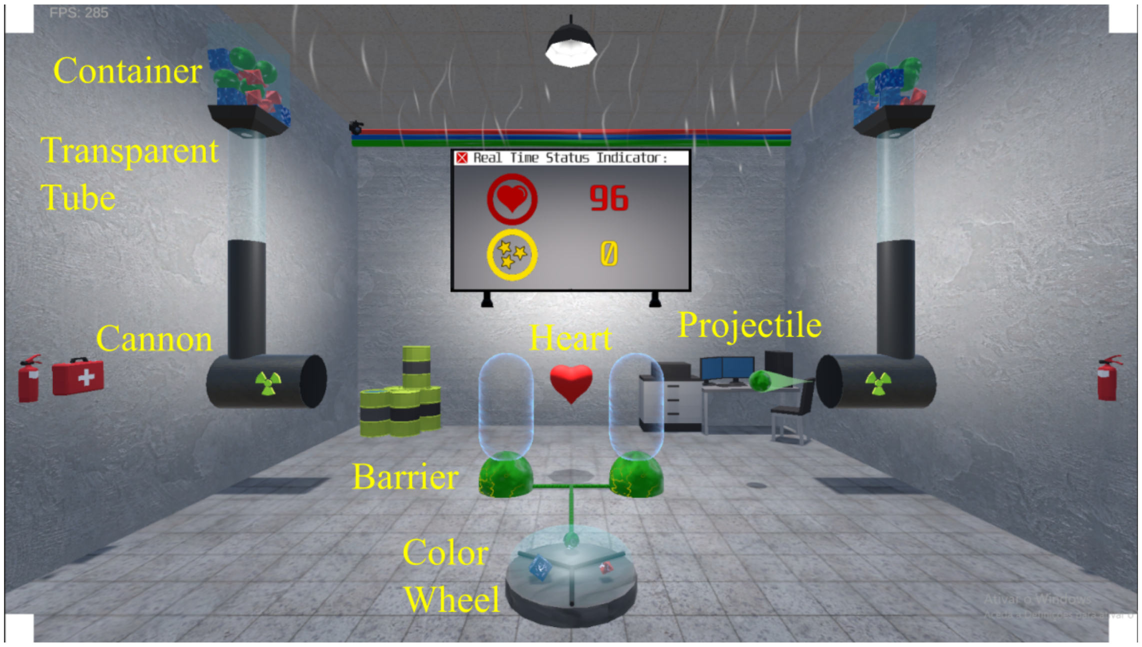

Figure 8. Game's screenshot with text overlaid in yellow.

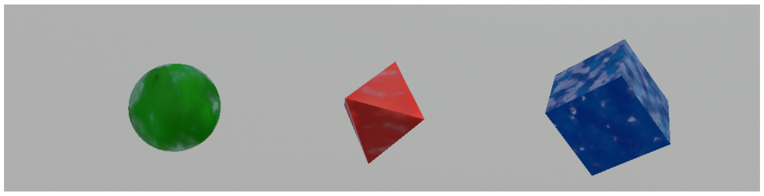

Figure 9. Different projectiles.

make the player lose health. For each projectile that reaches the ceiling the player is rewarded with 100 score points, for each projectile that hits the heart, the player is penalized 50 score points.

The different types of projectiles are an extra aspect that requires the player to be attentive. The barrier and projectile types must match in order to not explode the projectiles on impact. If this happens the projectiles can never reach the ceiling and therefore cannot score any points. As the projectiles fall through a transparent pipe, the player must select the correct type of protection barrier for the falling projectile, as depicted in Fig. 11. The barrier and the projectile correspond when they have the same color. To do this, the player must rotate a color wheel present under the protection barriers, by pressing the left and right arrow keys, on the keyboard.

The game runs on a loop. First a projectile type is randomized in the cannon on one side and the projectile falls through the transparent pipe. Then, after a random amount of time that projectile is

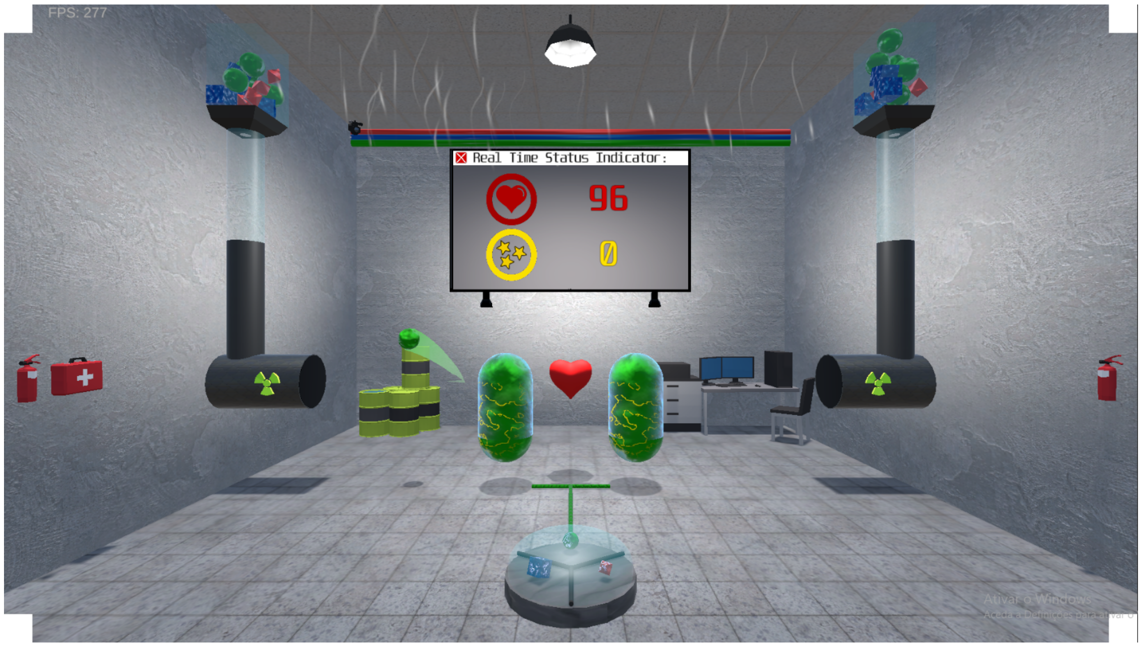

Figure 10. Game's screenshot - reflecting a projectile. The barrier deflected the projectile away from the heart, by intercepting its trajectory.

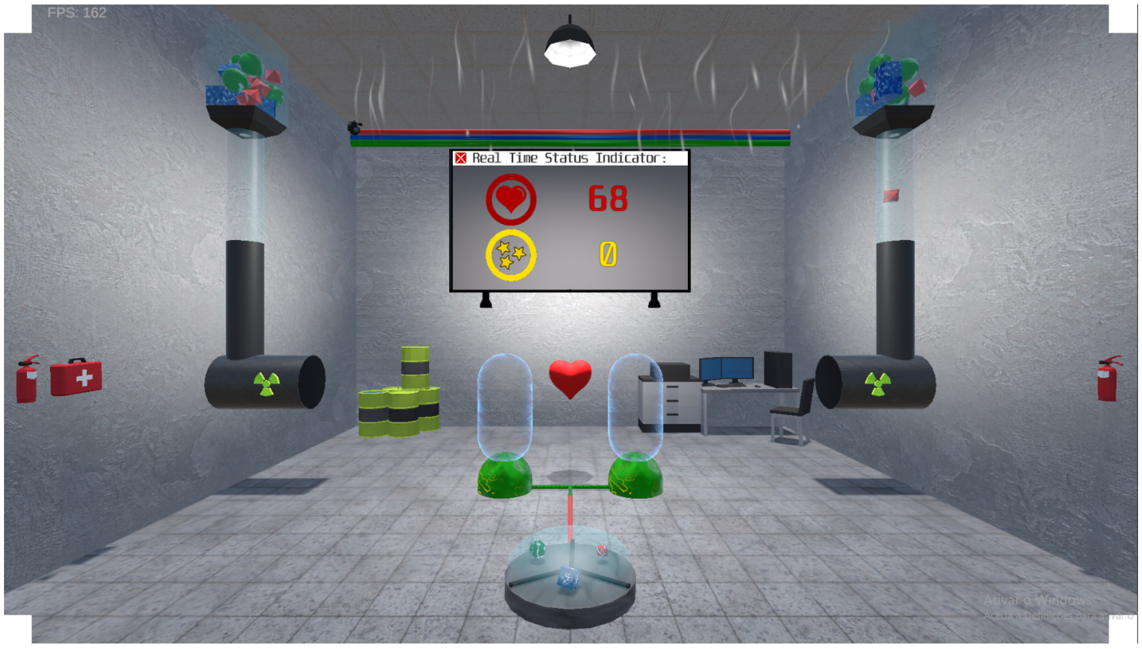

Figure 11. Game's screenshot - selecting barrier type. The player identifies the type of projectile falling through the pipe and rotates the color wheel in order to select the matching barrier.

launched. Whenever that projectile explodes against a barrier, hits the heart or reaches the ceiling, the same process initiates on the other side. This cycle is repeated for the duration of the gameplay, always alternating between the left and right side.

The game ensures that the same projectile type can never be launched in succession; for instance, if a green projectile was launched from the left, then it would not be possible for that projectile type to be fired again on the right side in the next cycle.

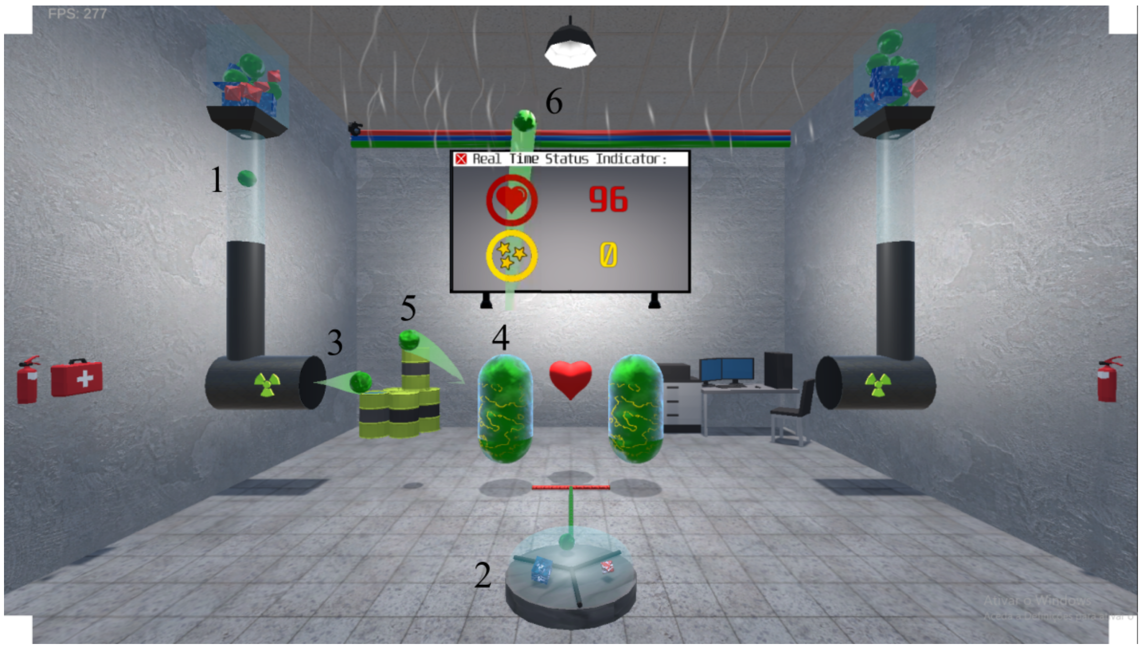

Figure 12. Game's cycle. This image was created for exemplification purposes, it is not an actual game's screenshot.

Figure 12 illustrates the game play loop with the intended interaction from the player. This image is a combination of several moments extracted from gameplay put together for explanation purposes, a situation that could never occur during a gaming session. As explained, a projectile falls through the pipe, (1), then the player selects the corresponding color for the barrier, (2). The projectile is then launched from the cannon, (3), the player reacts by raising the barrier (4) and finally the projectile is bounced away from the heart, (5,6). The projectile does not reach the ceiling and therefore the player does not score any points, (5). The projectile reaches the ceiling earning points (6). The same would then occur on the opposite side of the room.

4.2. Game's dynamic difficulty adjustment

The game's difficulty is defined by the speed at which the projectiles are launched. Lowering the speed not only gives more time for the player to select the color of the barriers as well as it increases the reaction window that the players have to reflect the projectile to the ceiling. The ability to set a dynamic difficulty adjustment in a game can create a better experience for the players by avoiding boredom or frustration and also increase the player's immersion [10].

In order to provide a good experience for the player, the game observes the players with the purpose of identifying the best difficulty for each player. To calculate the best speed for the projectiles, the players must go through several segments, with each one having a certain projectile speed. A segment is defined as a window of gameplay that changes its difficulty level according to the player's gameplay. Each segment has a duration of one minute, and there is a total of five segments. The projectile speed s_p of a segment n is empirically defined as:

$$s_p(n) = 100 + 150 \cdot n. \tag{4.1}$$

In this manner, it is possible to increase the speed significantly and consistently for each of the segments. It also allows to calculate a speed according to each segment, or even more precise values. That is to say that if the intended speed for the player is the one in between segment 1 and 2, the value 1.5 can be used to compute the speed.

During these segments, there is another speed adjustment of a smaller magnitude and is dependent on the success of the player. For instance, if the player scores some points, then the speed increases slightly, on the other hand, if the player does not score, then the speed decreases in the same amount.

While the player is in these segments, the game keeps count of the number of times the heart gets hit, h, and the times that it does not, q. This is used after each segment n to calculate the success ratio of that segment, $r_n = \frac{h}{q}$. To keep an interesting balance between skill and challenge, the preferred success ratio was empirically defined to be 1, which corresponds to when the players succeed as much they fail.

For each segment n a deviation value of the ratios to 1 is computed, $d_n = |1 - r_n|$. After the player plays five segments, the game computes the weighted average of the segments according to the deviation values, as follows:

$$\phi(k) = \frac{\sum_{n=1}^{k} n \cdot d_n}{\sum_{n=1}^{k} d_n}, \tag{4.2}$$

where k is the number of segments and d_n represents the deviation for segment n.

The weighted average corresponds to the segment in which the projectile speed is more adequate. For instance, $\phi(k) = 2$ corresponds to the second segment. This value may be a real number, $\phi(k) \in R$, and so it may represent a position in between the number of the segments. For example, if $\phi(k) = 1.5$, then it means that the closer segments for the player are segment 1 and segment 2. This also means that the values in between segment 1 and segment 2 can be used to compute intermediate speed values. To calculate the preferred projectile speed for the player, the game computes $s_p(\phi(k))$. From this point on the speed is constant for the rest of the game.

This method to compute the adequate speed has advantages and disadvantages. The main advantage is that it allows the player to always have a challenge, since the speed is usually higher than the success ratio of 1. This gives the player the opportunity to improve and still not be able to perfect the game. The disadvantage that this method presents is that if the players are not able to have a good success ratio in the easier segments, the game would select a speed that would still be too fast for those players.

4.3. Game technical implementation

The game was developed using the Unity game engine with the purpose of being exported as a WebGL build. For the game to be played remotely, a local machine hosted the website to which the players would connect to play the game.

To simplify the players' procedure, the gathered data from the game was sent automatically to the local machine using php. To clarify, the game was played on the browser of each participant, since the game runs solely on the client's, participant, side. The only in-game usage of the connection to the server was to send the gathered data. Sending the data was handled using coroutines, which means that the game would run normally, even if sending the data was taking some time.

The UML diagram of the interactions between the different game objects in the game is depicted in Fig. 13. This diagram shows how the core of the game was designed and implemented. For example, each projectile type is implemented as a *prefab*, which is an object that can be instantiated into the game and deleted several times without much computational effort.

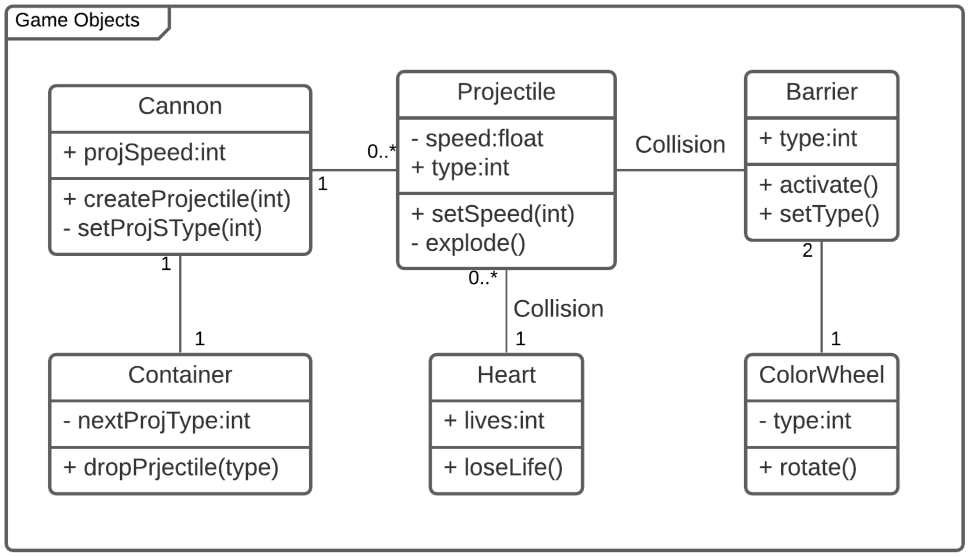

Figure 13. Game object's interactions. Each container is connected to one cannon which can fire several projectiles with different speeds. The projectiles can collide with the heart or with the barrier. The color wheel can change the type of the barrier.

Unity also makes the detection of collisions and other events easy and allows to customize these with different behaviours; for instance, the projectiles would explode when colliding with a barrier if the types did not match, or if the types matched, the projectiles would be reflected. Reflecting a projectile was also fairly easy to implement by using the built in physics engine, which provides for a more realistic interaction between both game objects. Aside from these game objects there are other more complex classes that handle the gameplay loop, the different rounds, the game adaptation, and other parts of the game implementation.

The game was developed over several months and with some gameplay testing sessions recurring to two participants. In a later stage of development, the game was tested by another subject. These testing sessions provided useful feedback from a player perspective, which improved the game's experience. These testing sessions also allowed for the testing of the game through the website, as well as to find bugs.

CHAPTER 5

Experimental Protocol

This chapter describes in detail how the proposed system for detecting divided attention was implemented and integrated in the game and how the experimental protocol was designed and applied in individual testing sessions.

5.1. System Integration

As explained, one of the purposes of the game is to be able to test the system's capabilities of detecting divided attention and to identify which are the best features to detect it. This analysis is done mostly off-line, which means that a part of the system is directly implemented in the game and another part is not. For the purposes of experimental validation of the proposed system, the information gathering module is the only module integrated in the game. This enables a more in depth interpretation of each step of this system. Section 3.5 describes the methodology that the system should follow to be to be actually deployed in a real-life game design and development pipeline.

Table 2. In-game events and reaction events correspondence

		In-game events	
		new projectile event	launch projectile event
Reaction events	Change barrier type	X	
	Raise barriers		X

In the developed game there are two game events, the first is when the game selects the next projectile type, hereafter called of *new projectile event*, and the second is when the projectile is launched, hereafter called *launch projectile event*. The player can react to these game events with the two related reaction events, showed in Table 2. As described in Chapter 4.1, the players must react to the new projectile event by selecting the protection barrier color, which is the corresponding reaction event. Similarly the player must also react to the launch projectile event by activating the protection barriers at a precise moment, which is the corresponding reaction event.

These events allow for the creation of several features to measure the player's performance in the game. According to the system integration methodology (see Section 3.5) the features are identified and classified as direct performance features or as inverse performance features. The direct performance features are:

- the number of times the player reacts on-time to the in-game events;
- the score of the player.

The inverse performance features are:

- the number of times the player does not react to the in-game events;
- the number of times the player reacts early to the in-game events;
- the number of times the player reacts too late to the in-game events;
- mean reaction time to the in-game events;
- standard deviation of the reaction times to the in-game events;
- the number of times that the barrier is activated and does not match the projectile type;
- the number of lives lost by the player.

5.1.1. No, early, on-time, and late reactions

The no reaction is defined to be when the player does not provide a valid input that could affect the current in-game event. For example, if the person changes the color of the barriers after the launch projectile event, then the person did not react accordingly and, thus, this event is counted as a no reaction. The same concept applies to the new projectile event.

The early reactions are reactions that are related to the event but are performed before the ideal moment, for example, raising the barriers too early and not protecting the heart (see Fig 14), zone A. It is not possible to react early to the new projectile event, as the event occurs right after the previous projectile explodes. Also the players would instinctively change the color after realising that the color of the projectiles never repeats consecutively.

The on time reactions are the ones that allow the player to score; in the case of the new projectile event, these are the moments in which the player changes color before the barrier is raised. An on time reaction to the launch projectile event is when the projectile reaches the ceiling, scoring points (see Fig 14), zone B.

Finally, the late reactions are the reactions that are performed after the ideal moment; for the new projectile event it means that the player changed the color too late, and when raising the barrier it would not be changed to the new selected color. For the launch projectile event, it is when the player reacts after the moment that would reflect the projectile to the ceiling, (see Fig 14), zone C.

The different projectile speeds that the game selects for each player provided each player with different optimal timings to score. To clarify, slower projectile speeds offered more time for the player to react than faster speeds. This means that the area of each zone from Fig. 14 would be different for each projectile speed.

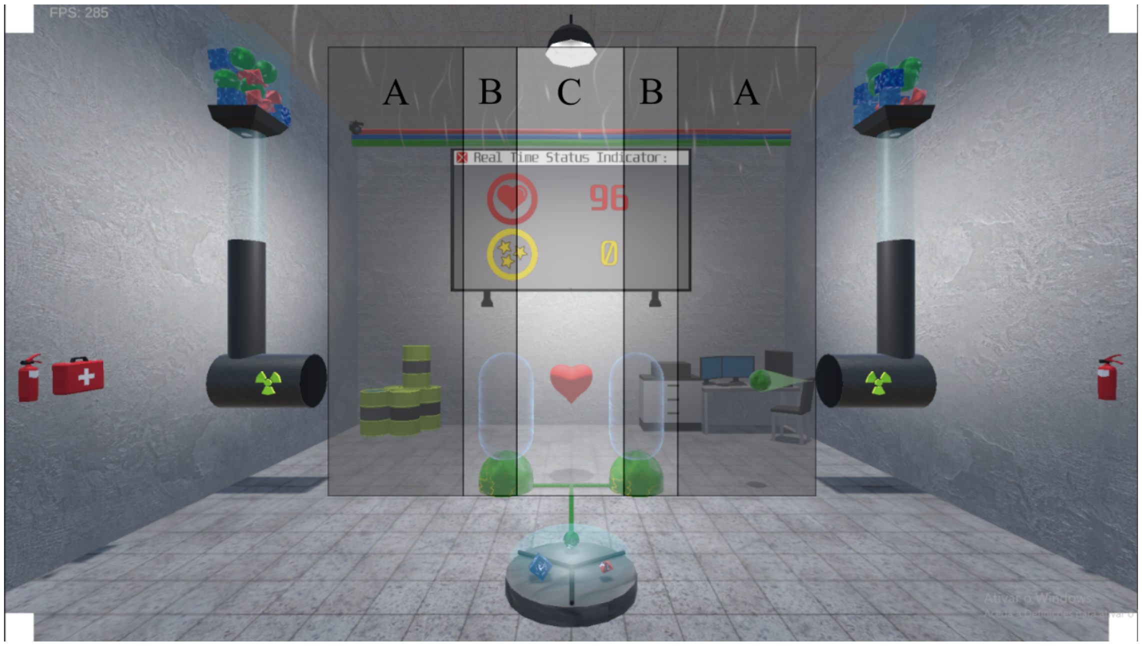

Figure 14. Game's screenshot with overlaid zones - Early reaction, zones A, on-time reactions, zones B and late reactions, zone C. These zones are different according to the speed of the projectile.

5.1.2. Mean and Standard Deviation of the reaction time

The reaction times are computed by an algorithm, which finds the correspondences between a game event, and a reaction event. For example, the new projectile event and rotating the color wheel, respectively. For this to happen the algorithm requires data from each game event and their related reaction events, which means the pair that is connected by the in-game logic. The game gathers this information by simply storing the events and their correspondent timestamp in a queue, E, as the game is played. The game events described before may not be connected to each other, e.g., if a player changes the color of the barrier instead of raising it, the algorithm considers that the player did not react to the game event, however when the game event and reaction match, then, the algorithm computes a reaction time for that game event. This enables the two events to have their own independent reaction times, which is better since it enables the reaction times to be more consistent. The recursive algorithm is presented in Fig. 15.

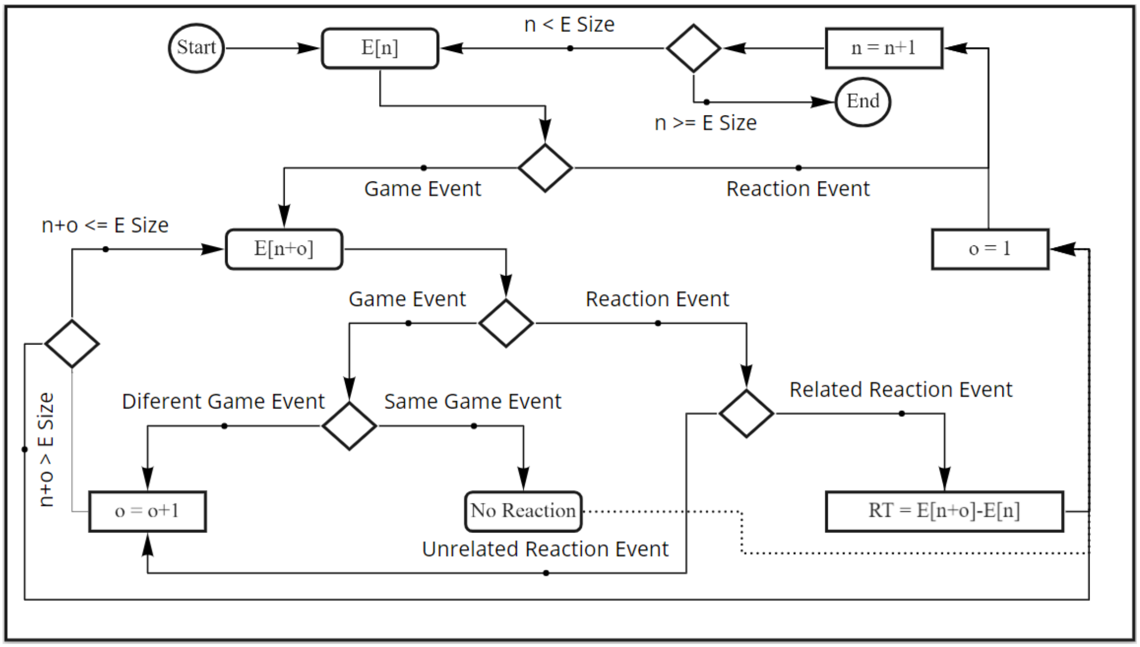

Figure 15. Algorithm's diagram for the pairing of in-game events and reaction events.

The algorithm processes the queue, E, where each event and correspondent timestamp are stored in chronological order for each time window. The algorithm can go through each element of the queue, $E[n]$, and find the correspondent elements to calculate the reaction time. For a given event $E[n]$ of the queue at position n, the algorithm identifies if $E[n]$ is either a game event or a reaction event. In the the case of being a game event, the algorithm searches across the subsequent elements of the queue for a corresponding reaction event $E[n + o], o >= 1$, (e.g., if $E[n]$ is launch projectile event and $E[n + o]$ is the reaction event from when the player blocked the projectile). The reaction time is computed as the difference between the timestamps associated to both events. If another game event of the same type of $E[n]$ is found before a corresponding reaction event, then the algorithm assumes that the player did not react to $E[n]$ and, so, it proceeds to the next element in the queue, by increasing the n value, $n \leftarrow n + 1$. The recursive algorithm ends when it reaches the last element of the queue.

The reaction times of each time window are exported into different files, which are then used to calculate the mean and the standard deviation for each time window.

5.1.3. Wrong Color, Score and Lost Lives

The wrong color feature is the number of times that the player activated the barrier with a color that did not match the projectile launched. Situations in which the player selected an incorrect color but did not raise the barrier are excluded from this feature. The score is computed as described in Chapter 4.1, concretely, is the value of score that the player got in that time window. The lost lives is the number of lives lost in a time window.

5.2. Secondary Tasks

In order to efficiently test the system, two secondary tasks were employed specifically to induce a divided attention state into the testers.

5.2.1. Counting Task

The first secondary task consists of having the testers count out loud, henceforth *counting task*. This task is purely cognitive, and in order for the task to present some challenge, the player must count from the number one hundred backwards in intervals of three. This task has been used in other research, mainly with the objective of simulating an extra cognitive workload during the execution of a main task [30]. Even though the task seems simple, it still forces the players to actively think about the next number. This is how the task is able to compete with the game for the players' cognitive resources.

5.2.2. Images Task

The second secondary task requires that the players look at some images and identify them, henceforth *images task*. This task is both cognitive and visual, and, therefore competes specifically with the player's perceptual resources. To clarify, the player has to count the number of circles that appear in four images, each one placed on a corner of the screen. The images are visible for a short duration of 2.5 seconds and reappear every t seconds, t being randomly chosen between 4 and 6. The images are also randomized form a pool of four geometric figures (see Fig. 16), each image only containing a single geometric figure. This can be seen in detail in Fig. 17.

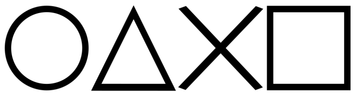

Figure 16. Figures that the players must identify.

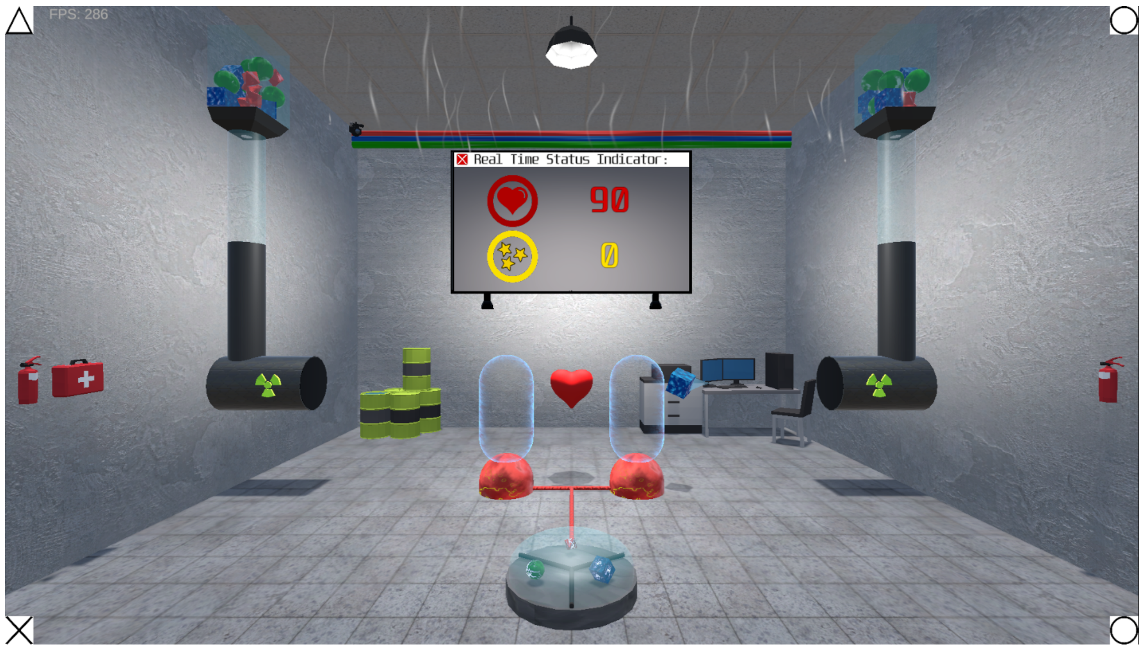

Figure 17. Game's screenshot - Images Task. The different geometric figures that the player must identify appear in the corners of the image.

5.3. Experimental Procedure

The experimental procedure starts with a small briefing of the experience, which is described in detail in the following section. Then, the player starts the experiment, which is split three sets, composed by multiple rounds, as depicted in Table 3. Each round is separated by a ten seconds countdown, visible in the screen, giving the player some preparation time, as depicted in Fig. 18.

The initial set has the main purpose of introducing the player to the game. This set is composed of two rounds, the first round is a tutorial round, where the player can understand the game mechanics and game cycle by playing it. To simplify the learning task, on this round the player is not

Table 3. Protocol Rounds Distribution

Set	**Initial Set**		**1st Set**				**2nd Set**			
Rounds	**1**	**2**	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**
Calculate Speed		X								
Gather Data			X	X	X	X	X	X	X	X
Game	X	X	X		X		X		X	
Game and Counting				X				X		
Game and Images						X				X

required to select the barrier type, as the game handles it automatically. Even though the player does not have to select the barrier type, the player gets an understanding of how and when to select it. The total duration of this round is two minutes. The second round of the initial set is the round used to find the adequate difficulty level for the player, see Section 4.2. The duration of this round is five minutes, one minute per each segment, which, as explained, is a gameplay segment in which the game difficulty changes according to the player's gameplay.

The two other sets are identical to each other and are composed of four rounds each. Each round corresponds to a time window of the system (see Section 3.2), which means that the features are tracked during each round, and that the system predicts the attention state of the player for each round. It is also in these four rounds that the player executes the described secondary tasks. In

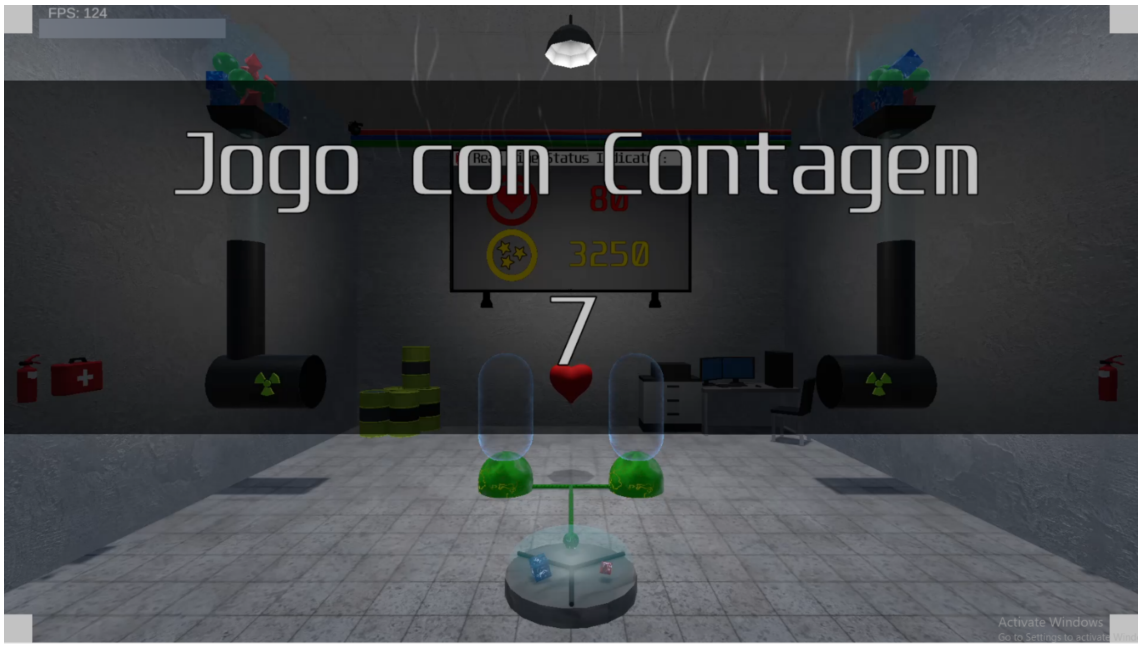

Figure 18. Game's screenshot - Countdown. The game is counting down to the next round.

the first and third rounds of these sets the player is induced a focused attention state, which means that the player is giving all attention to the game. In both the second and fourth rounds the player is induced a divided attention state, in other words, the player splits one's attention between some task and the game. This ensures that each set presents divided attention and focused attention. During the second round, the player is requested to execute the counting task while playing. During the fourth round the player is requested to execute the images task while playing. Each round has a duration of two minutes, for both the focused attention rounds as well as the divided attention rounds.

Each one of the sets is intended to represent a different moment in the system's implementation methodology (see Section 3.5). As described in the second phase of the implementation methodology, both the system and game must be tested in order to select the best features and combining methods, the first set being the one that emulates this situation. The second set is then used to emulate the usage of the game and system once the features and methods are selected. In other words, the first set is representative of sets gathered with controlled playtests executed while developing the game, whereas the second set is representative of sets gathered after the game being released to the market, that is, with actual players interacting with their own copy of the fully-developed game in uncontrolled settings.

5.4. Briefing

Before each testing session, participants went through a briefing, which consisted on explaining how the game worked as well as how the experiment would proceed. The secondary tasks were also briefly described, explaining the tasks simply but not providing all the details. For the counting task, the player is only told that the player must count out loud, however not mentioning the specific number and interval. This information is given during the countdown before each round. For the images in the corners, the players do not know which of the figures they must count, this information being given during the countdown before each round. It was explained to the participants that this was an experiment to test a system, not the participants. They were also informed that all the data gathered would be anonymous and if by any reason they changed their minds and decided to give up, all data would be deleted.

CHAPTER 6

Evaluation

This chapter presents the results obtained in the individual user testing sessions. Due to the COVID-19 pandemic, these sessions were conducted remotely and the game screen was shared. The game was hosted online with a WebGL build.

In total, 14 participants took part in this experiment, 11 males and 3 females, with an average age of 22.6 years and a standard deviation of 1.11. Eleven participants were college students, two had a full-time job and one participant was under aged, 14 years. Seven of the participants regularly played games, and the other seven only occasionally.

Considering that the testing sessions were done remotely, the participants played the game at their own setups, with different screen sizes, headphones, etc. Interestingly, the different setups can be seen as more realistic scenario, as if the game would have been released.

6.1. Prediction metrics

The accuracy, precision, recall, false positive rate and f1-score were calculated for each predicted set and for the combinations as well. In order to compute these values, it is required to compute the confusion matrices, which are composed of four possible prediction outcomes (see Table 4):

- True positive (TP): the system predicts a divided attention state in time windows that the player was induced into a divided attention state;
- False positive (FP): the system predicts a divided attention state in time windows that the player was induced into a focused attention state;
- True negative (TN): the system predicts a focused attention state in time windows that the player was induced into a focused attention state;
- False negative (FN): the system predicts a focused attention state in time windows that the player was induced into a divided attention state.

Accuracy, A, is used to evaluate the ability of the system to correctly predict any attention state. This metric is defined as the fraction of total correct predictions by the total number of predictions:

$$A = \frac{TP + TN}{TP + FP + TN + FN}. \tag{6.1}$$

Precision, P, is used to evaluate the system's credibility when predicting a divided attention state. Higher precision implies less false positives. This metric is defined as the fraction of correct divided attention predictions, TP, by the total of divided attention predictions:

$$P = \frac{TP}{TP + FP}. \tag{6.2}$$

Recall, also called true positive rate, TPR, is used to evaluate the ability of the system to correctly predict the divided attention state time windows. This metric is defined as the fraction of the number of correct divided attention predictions by the total cases of induced divided attention:

$$TPR = \frac{TP}{TP + FN}. \tag{6.3}$$

False positive rate, FPR, is used to evaluate the ability of the system to correctly predict focused attention state time windows. This metric is defined as the fraction of number of wrong divided attention predictions by the total of cases of induced focused attention state:

$$FPR = \frac{FP}{FP + TN}. \tag{6.4}$$

The f1-score, F_1, is used to combine the precision and recall and, therefore, evaluates the system's precision and recall at once. F_1 score is 1 if both precision and recall are 1, and 0 if either of the precision or recall is 0. This metric is defined as the harmonic mean of precision, P, and recall, TPR:

$$F_1 = 2 \cdot \frac{P \cdot TRP}{P + TPR}. \tag{6.5}$$

The data gathered by the system, specifically the different values of features for each time window and participant, were obtained and stored. These feature values can be seen in the Appendices A and B.

Table 4. Possible prediction outcomes.

		System's Prediction	
		Divided Attention	Focused Attention
Real Attention State	Divided Attention	True Positive	False Negative
	Focused Attention	False Positive	True Negative

As described in Section 5.3, the first set emulates the tests performed during the integration of the game in which the proposed system will be implemented, providing the results that allow the identification of the best features and combinations of predictions to be used. The reference value is computed with the same time windows that are being predicted. In other words, the system computes the reference value from a group of four time windows, i.e., the rounds from the first set. Then, the same group of time windows is predicted with that reference value. The three features that present the highest values of accuracy are the ones selected.

The second set emulates the usage of the final product, for example a released game. In this test, the reference value is computed from the time windows of the first set and used to predict the time windows of the second set. This implies that the reference value was computed using time windows (Set 1) that occurred before the predicted time windows (Set 2). This is the most realistic setting and should help understand if the selected features were good at detecting the attention state of the player.

Combining different predictions is addressed by combining pairs of features. The three features with highest accuracy are selected from the results of the first set and combined in pairs with each other, using the And and Or operations, (see Section 3.4). The two operations are employed so as to be able to compare the results from both and identify advantages and disadvantages. The combining of these features is conducted in both sets to see how they compare.

As described in the system's implementation methodology, the best features can be different depending on the response of the game and how it may affect the player. For the following tests, the best features are selected based on their accuracy. All the features are still tested when predicting the attention state for each time window of the second set, allowing to analyse the different metrics of all the features of the second set.

6.2. Sub-module: Version A - feature selection

This section presents the results of the predictions of the system with the version A of the sub-module for both sets. Version A of the sub-module computes the reference value based on two high performance time windows per feature (see Section 3.1).

6.2.1. First set

Table 5 contains the results of the attention state predictions for the time windows of the first set. As explained, this emulates the test that would be done during the implementation of the system in order to select the best features.

Four features presented an accuracy above 0.90. These features are the on time reactions to the launch projectile event, late reactions to the new projectile event, the wrong color and, the total score. Six other features presented an accuracy value between 0.80 and 0.90. Six of the features did not present any false positives. The worst features were the lives lost and the early reaction time to launch projectile events with an accuracy bellow 0.61. The three selected features, according to their accuracy are:

(1) The score with an accuracy of 0.95.
(2) The wrong color with an accuracy of 0.93.
(3) The on-time reactions to the launch projectile event with an accuracy of 0.93.

6.2.2. Second set

Table 6 contains results of the attention state predictions for the time windows of the second set, which, as described, is performed using the reference value computed from the first set. This emulates the usage of the system after being implemented into the game, as if it had been released. This would only use the selected features; nevertheless, as mentioned, it was decided to compute the metrics for the predictions of each feature.

Table 6 shows that there are no features that have an accuracy above 0.90; however, three present an accuracy above 0.80, which are the late reactions to the the new projectile event, the wrong color, and the total score. All features presented false positives, with the best ones having under 3 false positives. The worst features were the early reaction time to the launch projectile event, the mean of the reaction time to the launch projectile event and the lives lost. These features presented an

Table 5. Sub-module: Version A - reference value from Set 1 applied to Set 1.

	No Reaction				Early		On Time				Late			
	LP		NP		LP		LP		NP		LP		NP	
	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	23	5	22	6	10	18	24	4	22	6	15	13	23	5
Negative	1	27	0	28	4	24	0	28	0	28	3	25	0	28

Accuracy	0.89	0.89	0.61	0.93	0.89	0.71	0.91
Precision	0.96	1.00	0.71	1.00	1.00	0.83	1.00
Recall	0.82	0.79	0.36	0.86	0.79	0.54	0.82
FPR	0.04	0.00	0.14	0.00	0.00	0.11	0.00
F1	0.88	0.88	0.48	0.92	0.88	0.65	0.90

	Mean				ST-Dev				Wrong Color		Score		Lives Lost	
	LP		NP		LP		NP							
	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	13	15	18	10	22	6	24	4	24	4	25	3	24	4
Negative	2	26	1	27	1	27	3	25	0	28	0	28	23	5

Accuracy	0.70	0.80	0.88	0.88	0.93	0.95	0.52
Precision	0.87	0.95	0.96	0.89	1.00	1.00	0.51
Recall	0.46	0.64	0.79	0.86	0.86	0.89	0.86
FPR	0.07	0.04	0.04	0.11	0.00	0.00	0.82
F1	0.60	0.77	0.86	0.87	0.92	0.94	0.64

LP - launch projectile event
NP - new projectile event
P - positive
N - negative

Table 6. Sub-module: Version A - reference value from Set 1 applied to Set 2.

	No Reaction				Early		On Time				Late			
	LP		NP		LP		LP		NP		LP		NP	
	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	13	15	18	10	9	19	15	13	14	14	19	9	18	10
Negative	1	27	5	23	3	25	2	26	2	26	3	25	1	27

Accuracy	0.71	0.73	0.61	0.73	0.71	0.79	0.80
Precision	0.93	0.78	0.75	0.88	0.88	0.86	0.95
Recall	0.46	0.64	0.32	0.54	0.50	0.68	0.64
FPR	0.04	0.18	0.11	0.07	0.07	0.11	0.04
F1	0.62	0.71	0.45	0.67	0.64	0.76	0.77

	Mean				ST-Dev				Wrong Color		Score		Lives Lost	
	LP		NP		LP		NP							
	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	10	18	19	9	19	9	22	6	22	6	22	6	22	6
Negative	6	22	4	24	3	25	7	21	2	26	3	25	13	15

Accuracy	0.57	0.77	0.79	0.77	0.86	0.84	0.66
Precision	0.63	0.83	0.86	0.76	0.92	0.88	0.63
Recall	0.36	0.68	0.68	0.79	0.79	0.79	0.79
FPR	0.21	0.14	0.11	0.25	0.07	0.11	0.46
F1	0.45	0.75	0.76	0.77	0.85	0.83	0.70

LP - launch projectile event
NP - new projectile event
P - positive
N - negative

accuracy below 0.66. Analysing the three selected features, it is possible to see that the accuracy per featured decreased relatively to Set 1:

(1) The score's accuracy lowered from 0.95 to 0.84.

(2) The wrong color's accuracy lowered from 0.93 to 0.86.

(3) The on-time reactions to the launch projectile event's accuracy lowered from 0.93 to 0.73.

6.3. Sub-module: Version A - Combining features

To analyse if the combining of features could prove useful with version A of the sub-module, the best three pairs of features were combined, for both sets. The combining in the first set represents the results that would have been had during the feature selection phase of the system's implementation methodology (see Section 3.5). The combining in the second set emulates the results that would have been obtained after the system was implemented into the game, which is to say a released game.

The combining of features is only beneficial if the metrics improve over the uncombined features. Therefore the following sub-sections compare the combining features' metric with the prediction's metric of each individual feature.

The three selected features that were grouped in pairs are the standard deviation of the reaction time to the new projectile event, the wrong color, and the score.

6.3.1. First pair: Score and Wrong Color

Table 7 shows the results of combining of the standard deviation of the reaction time to the new projectile event with the wrong color feature. The combining of these two features improved the accuracy in the first set when using the Or operation to combine them, achieving an accuracy of 0.98, with no false positives. The And operation reduced the number of false positives, in the second set, to 0, which, as explained can be better depending on the use case.

6.3.2. Second pair: On-time reaction to the launch projectile event and Score

Table 8 contains the results of the combining of the wrong color and the on-time reactions to the launch projectile event. The combining of these two features maintained the accuracy in Set 2 using both the Or operation, with an accuracy of 0.86. The And operation reduced the number of false positives to 0, in the second set, once again.

Table 7. Score combined with wrong color.

	Score				Wrong Color				Score and Wrong Color							
									And				Or			
	Set1		Set2		Set1		Set2		Set1		Set2		Set1		Set2	
	P	N	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	25	3	22	6	24	4	22	6	22	6	19	9	27	1	25	3
Negative	0	28	3	25	0	28	2	26	0	28	0	28	0	28	5	23

Accuracy	0.95	0.84	0.93	0.86	0.89	0.84	0.98	0.86
Precision	1.00	0.88	1.00	0.92	1.00	1.00	1.00	0.83
Recall (TPR)	0.89	0.79	0.86	0.79	0.79	0.68	0.96	0.89
FPR	0.00	0.11	0.00	0.07	0.00	0.00	0.00	0.18
F1	0.94	0.83	0.92	0.85	0.88	0.81	0.98	0.86

P - positive
N - negative

Table 8. On-time reaction to the launch projectile event combined with wrong color.

	On-Time LP				Wrong Color				On-Time LP and Wrong Color							
									And				Or			
	Set1		Set2		Set1		Set2		Set1		Set2		Set1		Set2	
	P	N	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	24	4	15	13	24	4	22	6	21	7	18	10	27	1	24	4
Negative	0	28	2	26	0	28	2	26	0	28	0	28	0	28	4	24

Accuracy	0.93	0.73	0.93	0.86	0.88	0.82	0.98	0.86
Precision	1.00	0.88	1.00	0.92	1.00	1.00	1.00	0.86
Recall (TPR)	0.86	0.54	0.86	0.79	0.75	0.64	0.96	0.86
FPR	0.00	0.07	0.00	0.07	0.00	0.00	0.00	0.14
F1	0.92	0.67	0.92	0.85	0.86	0.78	0.98	0.86

P - positive
N - negative

Table 9. On-time reaction to the launch projectile event combined with score.

	On-Time LP				Score				On-time LP and Score							
									And				Or			
	Set1		Set2		Set1		Set2		Set1		Set2		Set1		Set2	
	P	N	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	24	4	15	13	25	3	22	6	23	5	14	14	26	2	23	5
Negative	0	28	2	26	0	28	3	25	0	28	2	26	0	28	3	25

Accuracy	0.93	0.73	0.95	0.84	0.91	0.71	0.96	0.86
Precision	1.00	0.88	1.00	0.88	1.00	0.88	1.00	0.88
Recall (TPR)	0.86	0.54	0.89	0.79	0.82	0.50	0.93	0.82
FPR	0.00	0.07	0.00	0.11	0.00	0.07	0.00	0.11
F1	0.92	0.67	0.94	0.83	0.90	0.64	0.96	0.85

P - positive
N - negative

6.3.3. Third pair: On-time reaction to the launch projectile event and Wrong Color

Table 9 contains the results of combining of the score with the on-time reactions to the launch projectile event. The combining of these two features improved the accuracy in both sets with the Or operation achieving 0.96 for the first set and 0.86 for the second. The And operation reduced the number of false positives to 2 in the second set.

6.4. Sub-module: Version A - Discussion

After analysing the results it is possible to conclude that the version A of the sub-module predicted the attention state of the player with high accuracy values in different features (above 0.80). When combining the different selected features it was possible to improve the accuracy, in some situations, specifically using the Or operation. The And operation provided the ability to reduce the false positives, at a small cost in accuracy.

6.5. Sub-module: version B - feature selection

This section presents the results of the predictions of the the version B of the sub-module for both sets. Version B of the sub-module computes the reference values based on the highest and lowest performance time windows per feature (see Section 3.2).

6.5.1. First set

Table 10 contains the results of the attention state predictions for the time windows of the first set. Similarly to the previous sub-module, this emulates the test that would allow the selection of the best performance features, during the implementation phase of the system.

There are five features which presented an accuracy above 0.90, these features being the wrong color, the standard deviation of the reaction times to the launch projectile event, the late reactions to the new projectile event, the on time reactions to the launch projectile event, and the score. Five other features presented an accuracy between 0.80 and 0.90. Three of the features did not show any false positives. The worst features were the early reaction to the launch projectile event and the lives lost, with accuracy below 0.57. The three selected features, according to their accuracy, are:

(1) The score with an accuracy of 0.93;
(2) The standard deviation of the reaction time to the launch projectile event with an accuracy of 0.93;
(3) The wrong color with an accuracy of 0.91.

6.5.2. Second set

Table 11 contains the results of the attention state predictions for the second set, which, as described, is performed using the reference value computed from the first set. Once again, this emulates the usage of the system after being implemented into the game. The metrics were computed for all the features, to allow for the comparison of those features between both sets.

Table 11 shows that there is one feature with an accuracy above 0.90, which is the wrong color. Three features presented no false positive predictions and three other features that presented an accuracy in the 0.80 to 0.90 interval. The worst features are the early reaction to the launch projectile events, the mean reaction time to the launch projectile event, the score and the lives lost, all of these

Table 10. Sub-module: Version B - reference value from Set 1 applied to Set 1.

	No Reaction				Early		On Time				Late			
	LP		NP		LP		LP		NP		LP		NP	
	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	24	4	23	5	20	8	23	5	22	6	25	3	26	2
Negative	3	25	1	27	16	12	0	28	0	28	10	18	3	25

Accuracy	0.88	0.89	0.57	0.91	0.89	0.77	0.91
Precision	0.89	0.96	0.56	1.00	1.00	0.71	0.90
Recall	0.86	0.82	0.71	0.82	0.79	0.89	0.93
FPR	0.11	0.04	0.57	0.00	0.00	0.36	0.11
F1	0.87	0.88	0.63	0.90	0.88	0.79	0.91

	Mean				ST-Dev				Wrong Color		Score		Lives Lost	
	LP		NP		LP		NP							
	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	26	2	27	1	28	0	26	2	27	1	24	4	26	2
Negative	12	16	10	18	4	24	4	24	4	24	0	28	24	4

Accuracy	0.75	0.80	0.93	0.89	0.91	0.93	0.54
Precision	0.68	0.73	0.88	0.87	0.87	1.00	0.52
Recall	0.93	0.96	1.00	0.93	0.96	0.86	0.93
FPR	0.43	0.36	0.14	0.14	0.14	0.00	0.86
F1	0.79	0.83	0.93	0.90	0.92	0.92	0.67

LP - launch projectile event
NP - new projectile event
P - positive
N - negative

presenting an accuracy below 0.64. Analysing the three selected features, it is possible to see that the accuracy decreased relatively to Set 1:

(1) The wrong color's accuracy decreased from 0.93 to 0.91;
(2) The standard deviation of the reaction time to the launch projectile event's accuracy decreased from 0.93 to 0.79;
(3) The score's accuracy decreased form 0.91 to 0.64.

6.6. Sub-module: Version B - Combining features

To analyse if the combining of features could improve the predictions using the version B of the sub-module, the best three pairs of features were combined, for both sets. As explained, the combining of the first set represents the results that would have been obtained during the second phase of the system's implementation methodology. The combining of the second set emulates the tests that would have been done after the system was implemented into the game.

As described before, the combining of features is only beneficial if the metrics improve over the uncombined features. Therefore the following sub-sections compare the combining features' metric with the prediction's metric of each individual feature.

The three selected features are the wrong color, the standard deviation of the reaction times to the launch projectile event, and the score.

6.6.1. First pair: Standard deviation of the reaction time to the launch projectile event and Score

Table 12 contains the results of combination of the standard deviation of the reaction times to the launch projectile event and the score. The combining of these two features did not improve the accuracy of the predictions. The And operation maintained the number of false positives reducing the accuracy in the second set.

6.6.2. Second pair: Wrong Color and Score

Table 13 contains the results of combining of the wrong color and the score. The combining of these two features did not improve the accuracy in any of the sets. The And operation maintained the number of false positives but reduced the accuracy significantly in the second set.

Table 11. Sub-module: Version B - reference value from Set 1 applied to Set 2.

	No Reaction				Early		On Time				Late			
	LP		NP		LP		LP		NP		LP		NP	
	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	16	12	20	8	18	10	12	16	15	13	22	6	19	9
Negative	0	28	1	27	12	16	1	27	0	28	7	21	1	27

Accuracy	0.79	0.84	0.61	0.70	0.77	0.77	0.82
Precision	1.00	0.95	0.60	0.92	1.00	0.76	0.95
Recall	0.57	0.71	0.64	0.43	0.54	0.79	0.68
FPR	0.00	0.04	0.43	0.04	0.00	0.25	0.04
F1	0.73	0.82	0.62	0.59	0.70	0.77	0.79

	Mean				ST-Dev				Wrong Color		Score		Lives Lost	
	LP		NP		LP		NP							
	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	22	6	25	3	20	8	24	4	24	4	8	20	23	5
Negative	19	9	9	19	4	24	7	21	1	27	0	28	16	12

Accuracy	0.55	0.79	0.79	0.80	0.91	0.64	0.63
Precision	0.54	0.74	0.83	0.77	0.96	1.00	0.59
Recall	0.79	0.89	0.71	0.86	0.86	0.29	0.82
FPR	0.68	0.32	0.14	0.25	0.04	0.00	0.57
F1	0.64	0.81	0.77	0.81	0.91	0.44	0.69

LP - launch projectile event
NP - new projectile event
P - positive
N - negative

Table 12. Standard deviation of the reaction time to the launch projectile event combined with score.

	STDEV- LP				Score				On-time LP and Score							
									And				Or			
	Set1		Set2		Set1		Set2		Set1		Set2		Set1		Set2	
	P	N	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	28	0	20	8	24	4	8	20	24	4	7	21	28	0	21	7
Negative	4	24	4	24	0	28	0	28	0	28	0	28	5	23	5	23

Accuracy	0.93	0.79	0.93	0.64	0.93	0.63	0.91	0.79
Precision	0.88	0.83	1.00	1.00	1.00	1.00	0.85	0.81
Recall (TPR)	1.00	0.71	0.86	0.29	0.86	0.25	1.00	0.75
FPR	0.14	0.14	0.00	0.00	0.00	0.00	0.18	0.18
F1	0.93	0.77	0.92	0.44	0.92	0.40	0.92	0.78

P - positive
N - negative

Table 13. Wrong color combined with score.

	Wrong Color				Score				Wrong Color and Score							
									And				Or			
	Set1		Set2		Set1		Set2		Set1		Set2		Set1		Set2	
	P	N	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	27	1	24	4	24	4	8	20	23	5	8	20	28	0	24	4
Negative	4	24	1	27	0	28	0	28	0	28	0	28	4	24	1	27

Accuracy	0.91	0.91	0.93	0.64	0.91	0.64	0.93	0.91
Precision	0.87	0.96	1.00	1.00	1.00	1.00	0.88	0.96
Recall (TPR)	0.96	0.86	0.86	0.29	0.82	0.29	1.00	0.86
FPR	0.14	0.04	0.00	0.00	0.00	0.00	0.14	0.04
F1	0.92	0.91	0.92	0.44	0.90	0.44	0.93	0.91

P - positive
N - negative

Table 14. Wrong color combined with the standard deviation of the reaction time to the launch projectile event.

	Wrong Color				ST-DEV LP				Wrong Color and ST-DEV LP							
									And				Or			
	Set1		Set2		Set1		Set2		Set1		Set2		Set1		Set2	
	P	N	P	N	P	N	P	N	P	N	P	N	P	N	P	N
Positive	27	1	24	4	28	0	20	8	27	1	17	11	28	0	27	1
Negative	4	24	1	27	4	24	4	24	0	28	1	27	8	20	4	24

Accuracy	0.91	0.91	0.93	0.79	0.98	0.79	0.86	0.91
Precision	0.87	0.96	0.88	0.83	1.00	0.94	0.78	0.87
Recall (TPR)	0.96	0.86	1.00	0.71	0.96	0.61	1.00	0.96
FPR	0.14	0.04	0.14	0.14	0.00	0.04	0.29	0.14
F1	0.92	0.91	0.93	0.77	0.98	0.74	0.88	0.92

P - positive
N - negative

6.6.3. Third pair: Wrong Color and standard deviation of the reaction time to the launch projectile event

Table 14 contains the results of the combining of the wrong color and the standard deviation of the reaction times to the launch projectile event. The accuracy was maintained by using the Or operation in the second set. The And operation reduced the number of false positives to 0 in the first set, and maintained it on the second set.

6.7. Sub-module: Version B - Discussion

After analysing the results obtained when using the version B of the sub-module, it is possible to conclude that the system predicted the attention state of the players with a acceptable accuracy. When combining the selected features, the accuracy did not improve and the number of false positives also were not reduced significantly.

6.8. Discussion

The system was successful at predicting the attention state of the players using both versions of the sub-module, however each one presented some advantages and disadvantages. Version A of the sub-module presented a larger number of features with an accuracy above 0.90 than version B. Version A was also more successful at combining features, in some cases, reducing the false positives to zero. On the other hand, version B had a higher accuracy in some features in the second set in comparison to version A. For instance, for the second set, version A's maximum accuracy value was 0.86, while version B presented a maximum accuracy of 0.91 for that same set.

Overall, version A seems a better approach regarding various features in general, since the selected features lowered their accuracy from the first set to second by a maximum of 0.20. For version B, the accuracy of the selected features was lowered by a maximum of 0.27.

Version B may be better for certain specific features, since the difference in accuracy between the first and second sets for the wrong color was 0.02, whilst in version A, the minimum accuracy difference from the selected features between the first and second sets is 0.07.

Out of all pairs of predictions, the one with highest accuracy is the combination of the wrong color and score, which was computed with the reference values from the version B of the sub-module. This pair combined with the Or operation presented an accuracy of 0.91, and a false positive rate of 0.04. For this game, this is the best sub-module and features to predict the attention state of the player in the second set.

CHAPTER 7

Conclusions

The literature review showed that the attention state has been studied among different topics, however only a few articles mentioned detecting the attention state of the player using solely indirect feedback. The proposed system is versatile and only used indirect feedback. The performed evaluation of the system is an attempt at addressing the absence of a similar system designed for specifically for videogames.

The proposed divided attention detection system's main objective is to detect moments in which the player is in a divided attention state. The attention state of the player is predicted by measuring the player's performance throughout the gameplay, specifically by tracking indirect features of the game. This allows for the system to be more versatile as it can be implemented in different types of games. It is composed of modules and sub-modules, which makes it easier to comprehend and implement, as each module has its own function. The system's sub-module can have different versions to adapt the system's predictions according to different games and or features that may be more complex.

A game to benchmark the proposed system was designed and developed to not only provide a good experience for the players, but also present a reasonable amount of features for the system to use. The game was a short experience that was easily understood by the players, since the tutorial round simplified the learning process. The game also presented a dynamic difficulty adjustment algorithm, which provided for a more custom difficulty setting. This difficulty setting allowed the players to be presented with enough challenge so as to not become a boring experience, even considering that the players may improve their gameplay significantly. There were a total of nine features present in the game that were used by the system, six of which were independent per in-game event. In other words, the six features enabled each game event to make a prediction. The relation of the simplicity of the game with the number of extracted features demonstrates that more complex games can easily find and make use of a larger number of features, which could even improve the system's detection accuracy.

An implementation methodology was also designed and employed in order to guide developers that intend on implementing the proposed system. The methodology is composed of three phases, splitting the implementation process into simpler tasks. This implementation process was used when implementing the system into the developed game.

To induce the two different attention states, focused attention and divided attention, two secondary tasks were implemented. The divided attention state required players to execute one of the secondary tasks. The counting task competes with the player's cognitive resource and the images task competes with the player's visual attention resource.

The conducted user testing sessions, with a total of 14 participants, resulted in an analysis of the system's capabilities, using two different prediction sub-module versions, each version with their own advantages and disadvantages. Version A of the sub-module presented high accuracy values, and the combining improved the results slightly. Version B of the sub-module presented, overall, lower accuracy values than sub-module A. However, the best version to use in this game is version B due to the higher accuracy and lower false positive rate obtained.

Some of the features used were substantially better than others, having almost all the time windows correctly predicted. For instance, the wrong color feature with a minimum accuracy of 0.86 over all time windows and from both sub-modules. Some other features did not predict the attention state of the player with a good accuracy. For instance, the early reaction time to the launch projectile event, which presented a maximum accuracy value of 0.61 over all time windows for both sub-modules. Possibly, the system may be able to use these low accuracy features with another sub-module in an attempt to improve their accuracy.

Overall, the testing sessions showed promising results, with several features presenting high accuracy values (above 0.80) and low false positive rates (below 0.11).

7.1. Future Work

The divided attention detection system showed very interesting results. Nevertheless, there are still some limitations that should be addressed in future work. One of the limitations that this system presents is that the detection occurs only in a predetermined time window intervals.

Another improvement that could be implemented in the system is to experiment versions of the modules and sub-module different from the ones presented. For instance, the final prediction module could be improved upon using another method to combine the different predictions. For instance, using the difference between the feature value and the reference value, to compute weights and combine those predictions according to their weights. One last point of improvement for the system is to have it automatically select the best features according to each player, which will most likely improve the system's ability to detect divided attention significantly.

The proposed divided attention detection system does not depend on machine learning techniques to operate, which means that its dependency on historical data is low and its computational footprint. This is expected to reduce its development and runtime overhead. Conversely, including machine learning into the proposed system could help automate feature selection and achieve higher prediction accuracy. Finding the right amount of machine learning in the proposed system should be addressed in future work.

The system would benefit from being tested with more users and in a realistic setting, once the system is fully implemented and deployed in a game. Implementing the system in different games would support the versatility of the system even more. For example, indie games and other simpler games could use the system, also possibly improving the overall structure of the system, becoming more robust and better overall.

References

[1] A. Z. Abbasi, S. Nisar, U. Rehman, and D. H. Ting, “Impact of hexaco personality factors on consumer video game engagement: A study on esports,” *Frontiers in psychology*, vol. 11, pp. 18–31, 2020.

[2] J. K. Mullins and R. Sabherwal, “Gamification: A cognitive-emotional view,” *Journal of Business Research*, vol. 106, pp. 304–314, 2020.

[3] B. Teles, P. Mariano, and P. Santana, “Game-like 3d visualisation of air quality data,” *Multimodal Technologies and Interaction*, vol. 4, no. 3, p. 54, 2020.

[4] H. W. M. van den, I. W. A., and K. Y. A. de, “Effects of sensory immersion on behavioural indicators of player experience: Movement synchrony and controller pressure,” in *Proceedings of the 2009 DiGRA International Conference: Breaking New Ground: Innovation in Games, Play, Practice and Theory*, Brunel University, Sep. 2009.

[5] G. N. Yannakakis, P. Spronck, D. Loiacono, and E. André, “Player Modeling,” *Dagstuhl Follow-Ups*, vol. 6, pp. 45–59, 2013.

[6] B. Cowley, D. Charles, M. Black, and R. Hickey, “Toward an understanding of flow in video games,” *Computers in Entertainment (CIE)*, vol. 6, no. 2, pp. 1–27, 2008.

[7] J. M. McDowd, “An overview of attention: Behavior and brain,” *Journal of Neurologic Physical Therapy*, vol. 31, no. 3, pp. 98–103, 2007.

[8] R. Lara-Cabrera and D. Camacho, “A taxonomy and state of the art revision on affective games,” *Future Generation Computer Systems*, vol. 92, pp. 516–525, 2019.

[9] E. A. Boyle, T. M. Connolly, T. Hainey, and J. M. Boyle, “Engagement in digital entertainment games: A systematic review,” *Computers in human behavior*, vol. 28, no. 3, pp. 771–780, 2012.

[10] A. Denisova and P. Cairns, “Player experience and deceptive expectations of difficulty adaptation in digital games,” *Entertainment Computing*, vol. 29, pp. 56–68, 2019.

[11] M. Kim and Y. Y. Doh, “Computational modeling of players’ emotional response patterns to the story events of video games,” *IEEE Transactions on Affective Computing*, vol. 8, no. 2, pp. 216–227, 2017.

[12] A. Tato, R. Nkambou, and R. Ghali, “Towards predicting attention and workload during math problem solving,” in *Proceedings of the International Conference on Intelligent Tutoring Systems*, Springer, 2019, pp. 224–229.

[13] W. Wan, X. Cui, Z. Gao, and Z. Gu, “Frontal eeg-based multi-level attention states recognition using dynamical complexity and extreme gradient boosting,” *Frontiers in Human Neuroscience*, vol. 15, 2021.

[14] J. A. McKanna, H. Jimison, and M. Pavel, “Divided attention in computer game play: Analysis utilizing unobtrusive health monitoring,” in *Proceedings of the 2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, IEEE, 2009, pp. 6247–6250.

[15] Z. M. Osman, J. Dupire, S. Mader, P. Cubaud, and S. Natkin, “Monitoring player attention: A non-invasive measurement method applied to serious games,” *Entertainment Computing*, vol. 14, pp. 33–43, 2016.

[16] D. Shastri, A. Wesley, and I. Pavlidis, “Contact-free stress monitoring for user’s divided attention,” in *Human Computer Interaction*, IntechOpen, 2008, ch. 9.

[17] S. W. McQuiggan, S. Lee, and J. C. Lester, “Predicting user physiological response for interactive environments: An inductive approach.,” *AIIDE*, vol. 2006, pp. 60–65, 2006.

[18] J. Antunes and P. Santana, “A study on the use of eye tracking to adapt gameplay and procedural content generation in first-person shooter games,” *Multimodal Technologies and Interaction*, vol. 2, no. 2, p. 23, 2018.

[19] M. Lankes, S. Riegler, A. Weiss, T. Mirlacher, M. Pirker, and M. Tscheligi, “Facial expressions as game input with different emotional feedback conditions,” in *Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*, 2008, pp. 253–256.

[20] F. Schmiedek, M. Lövdén, and U. Lindenberger, “On the relation of mean reaction time and intraindividual reaction time variability.,” *Psychology and aging*, vol. 24, no. 4, pp. 841–857, 2009.

[21] L. Tamm, M. E. Narad, T. N. Antonini, K. M. O'Brien, L. W. Hawk, and J. N. Epstein, "Reaction time variability in adhd: A review," *Neurotherapeutics*, vol. 9, no. 3, pp. 500–508, 2012.

[22] T. N. Antonini, M. E. Narad, J. M. Langberg, and J. N. Epstein, "Behavioral correlates of reaction time variability in children with and without adhd.," *Neuropsychology*, vol. 27, no. 2, pp. 201–209, 2013.

[23] I. Penuelas-Calvo, L. K. Jiang-Lin, B. Girela-Serrano, D. Delgado-Gomez, R. Navarro-Jimenez, E. Baca-Garcia, and A. Porras-Segovia, "Video games for the assessment and treatment of attention-deficit/hyperactivity disorder: A systematic review," *European child & adolescent psychiatry*, pp. 1–16, 2020.

[24] M. H. Rahman and M. S. Islam, "Investigation of audio-visual simple reaction time of university athletes and non-athletes," *Journal Of Advances In Sports And Physical Education*, vol. 4, no. 3, pp. 24–29, 2021.

[25] P. Bickmann, K. Wechsler, K. Rudolf, C. Tholl, I. Froböse, and C. Grieben, "Comparison of reaction time between esports players of different genres and sportsmen," *International Journal of eSports Research (IJER)*, vol. 1, no. 1, pp. 1–16, 2021.

[26] S. Jose and K. Gideon Praveen, "Comparison between auditory and visual simple reaction times," *Neuroscience & Medicine*, vol. 1, pp. 30–32, 2010.

[27] C. D. Wickens, *Processing resources and attention*. CRC Press, 1984, pp. 63–97.

[28] K. Stojmenova and J. Sodnik, "Detection-response task—uses and limitations," *Sensors*, vol. 18, no. 2, p. 594, 2018.

[29] K. Stojmenova, G. Jakus, and J. Sodnik, "Sensitivity evaluation of the visual, tactile, and auditory detection response task method while driving," *Traffic injury prevention*, vol. 18, no. 4, pp. 431–436, 2017.

[30] S. C. Castro, D. L. Strayer, D. Matzke, and A. Heathcote, "Cognitive workload measurement and modeling under divided attention.," *Journal of experimental psychology: human perception and performance*, vol. 45, no. 6, p. 826, 2019.

[31] R. E. Miller, T. L. Brown, S. Lee, I. Tibrewal, G. G. Gaffney, G. Milavetz, R. L. Hartman, D. A. Gorelick, R. Compton, and M. A. Huestis, “Impact of cannabis and low alcohol concentration on divided attention tasks during driving,” *Traffic injury prevention*, vol. 21, no. 1, pp. 123–129, 2020.

[32] L. Itti and P. Baldi, “Bayesian surprise attracts human attention,” *Vision research*, vol. 49, no. 10, pp. 1295–1306, 2009.

[33] A. Alkhatlan and J. Kalita, “Intelligent tutoring systems: A comprehensive historical survey with recent developments,” *International Journal of Computer Applications*, vol. 181, pp. 1–20, 2018.

[34] S. Cetintas, L. Si, Y. P. P. Xin, and C. Hord, “Automatic detection of off-task behaviors in intelligent tutoring systems with machine learning techniques,” *IEEE Transactions on Learning Technologies*, vol. 3, no. 3, pp. 228–236, 2009.

[35] R. S. Baker, “Modeling and understanding students’ off-task behavior in intelligent tutoring systems,” in *Proceedings of the SIGCHI conference on Human factors in computing systems*, 2007, pp. 1059–1068.

[36] V. Beserra, M. Nussbaum, and M. Oteo, “On-task and off-task behavior in the classroom: A study on mathematics learning with educational video games,” *Journal of educational computing research*, vol. 56, no. 8, pp. 1361–1383, 2019.

[37] D. Carpenter, A. Emerson, B. W. Mott, A. Saleh, K. D. Glazewski, C. E. Hmelo-Silver, and J. C. Lester, “Detecting off-task behavior from student dialogue in game-based collaborative learning,” in *Proceedings of the International Conference on Artificial Intelligence in Education*, Springer, 2020, pp. 55–66.

Appendices

APPENDIX A

Tables with the performance features from Set 1.

Table 15. Features from gameplay from the first round of the Set 1.

Participant	No Reaction		Early	On Time		Late		Mean		ST-Dev		Wrong Color	Score	Lives Lost
	LP	NP	LP	LP	NP	LP	NP	LP	NP	LP	NP			
P1	0	0	9	23	44	9	2	0.65	1.16	0.08	0.31	1	2250	9
P2	0	0	6	37	48	5	0	0.51	0.78	0.04	0.74	1	3450	5
P3	1	0	8	36	47	4	1	0.55	0.91	0.05	0.12	0	3250	5
P4	1	0	6	28	44	13	3	0.46	0.91	0.11	0.24	2	2000	14
P5	2	0	5	35	47	6	1	0.52	0.7	0.05	0.31	4	2900	8
P6	0	0	8	41	51	2	0	0.46	0.94	0.03	0.18	0	4000	2
P7	0	0	5	38	46	6	1	0.47	1.04	0.08	0.19	4	3400	6
P8	0	0	9	40	50	2	0	0.45	0.63	0.02	0.21	0	3800	2
P9	0	0	4	38	48	6	0	0.57	0.63	0.25	0.39	0	3600	6
P10	1	0	7	35	46	4	1	0.52	0.8	0.04	0.21	0	3200	6
P11	0	0	15	32	49	3	0	0.45	0.26	0.05	0.19	1	3050	3
P12	0	0	7	34	50	9	0	0.45	0.9	0.06	0.13	3	2800	9
P13	0	0	3	44	49	1	0	0.49	0.53	0.03	0.25	3	4350	1
P14	0	0	5	38	47	5	0	0.54	1.13	0.04	0.15	3	3400	5

Table 16. Features from gameplay from the second round of the Set 1.

Participant	No Reaction		Early	On Time		Late		Mean		ST-Dev		Wrong Color	Score	Lives Lost
	LP	NP	LP	LP	NP	LP	NP	LP	NP	LP	NP			
P1	9	9	28	4	31	6	4	0.64	1.14	0.45	0.47	8	150	16
P2	3	1	11	25	44	9	3	0.51	0.79	0.05	0.48	6	1750	13
P3	6	2	6	22	34	8	7	0.6	1.13	0.09	0.42	5	1600	14
P4	18	8	7	10	30	13	8	0.54	1.05	0.23	0.51	8	0	32
P5	0	4	14	21	40	14	4	0.55	0.79	0.09	0.61	30	550	12
P6	1	17	14	27	23	5	6	0.5	1.24	0.07	0.52	25	1300	8
P7	2	2	15	21	41	11	3	0.55	1.2	0.39	0.39	15	1300	13
P8	11	8	5	25	30	7	6	0.51	1.08	0.1	0.6	6	1250	18
P9	2	0	9	30	46	7	2	0.52	0.52	0.05	0.46	4	2400	9
P10	5	0	10	28	42	4	3	0.5	1.08	0.06	0.4	15	1300	10
P11	4	10	16	25	39	5	1	0.45	0.98	0.06	0.62	10	1550	9
P12	14	7	4	10	33	21	3	0.55	1.13	0.12	0.55	5	0	37
P13	18	4	1	14	37	12	6	0.57	1.09	0.16	0.69	4	650	28
P14	3	11	8	24	26	12	6	0.66	1.22	0.41	0.59	16	1000	15

Table 17. Features from gameplay from the third round of the Set 1.

Participant	No Reaction		Early	On Time		Late		Mean		ST-Dev		Wrong Color	Score	Lives Lost
	LP	NP	LP	LP	NP	LP	NP	LP	NP	LP	NP			
P1	3	2	10	28	41	6	2	0.63	1.12	0.09	0.36	1	2400	9
P2	0	0	4	39	48	5	2	0.5	0.66	0.05	0.72	1	3600	6
P3	1	0	4	37	46	6	0	0.55	0.87	0.04	0.13	0	3350	7
P4	2	0	8	24	47	9	1	0.53	1.02	0.12	0.24	0	1750	16
P5	0	0	5	41	48	1	0	0.47	0.62	0.04	0.3	2	3950	1
P6	0	0	6	41	50	3	0	0.46	0.84	0.04	0.13	0	3950	3
P7	0	0	7	37	50	8	0	0.53	0.99	0.04	0.18	4	3100	7
P8	0	0	7	41	48	2	0	0.53	0.59	0.03	0.13	0	3900	2
P9	0	0	3	39	49	8	0	0.48	1.16	0.03	0.78	1	3700	6
P10	0	0	3	39	47	6	0	0.47	0.89	0.04	0.19	1	3600	6
P11	0	0	9	38	50	2	0	0.46	0.5	0.04	0.25	0	3600	4
P12	0	0	7	29	46	12	0	0.48	0.84	0.05	0.15	4	2450	12
P13	0	0	3	42	46	2	0	0.49	0.41	0.03	0.31	2	3800	2
P14	0	0	6	36	45	5	0	0.55	1.04	0.05	0.2	3	3250	5

Table 18. Features from gameplay from the fourth round of the Set 1.

Participant	No Reaction		Early	On Time		Late		Mean		ST-Dev		Wrong Color	Score	Lives Lost
	LP	NP	LP	LP	NP	LP	NP	LP	NP	LP	NP			
P1	8	2	13	13	42	13	5	0.63	1.24	0.16	0.39	4	250	21
P2	0	1	9	28	38	9	7	0.52	0.72	0.07	0.46	12	2350	9
P3	7	4	6	25	31	9	8	0.54	1.15	0.16	0.38	8	1600	16
P4	22	9	1	8	30	16	8	0.56	1.2	0.13	0.56	7	100	38
P5	0	0	7	25	45	11	1	0.5	0.82	0.12	0.42	4	1850	12
P6	3	4	5	29	32	13	8	0.49	1.24	0.1	0.47	13	1900	14
P7	1	7	9	29	36	10	4	0.5	1.15	0.09	0.42	11	1900	13
P8	1	0	4	36	46	8	3	0.49	0.83	0.05	0.41	7	3000	9
P9	0	1	6	37	47	5	0	0.53	0.84	0.06	0.58	5	3250	5
P10	2	2	9	33	45	3	2	0.52	1.05	0.12	0.35	5	3050	6
P11	2	2	9	32	46	7	0	0.48	0.68	0.06	0.4	11	2550	10
P12	7	5	6	21	38	15	4	0.48	1.04	0.11	0.37	4	1300	22
P13	4	0	3	33	46	9	4	0.52	0.77	0.09	0.45	1	2700	13
P14	9	7	2	27	36	9	1	0.59	1.11	0.06	0.6	5	1650	18

APPENDIX B

Tables with the performance features from Set 2.

Table 19. Features from gameplay from the first round of the Set 2.

Participant	No Reaction		Early	On Time		Late		Mean		ST-Dev		Wrong Color	Score	Lives Lost
	LP	NP	LP	LP	NP	LP	NP	LP	NP	LP	NP			
P1	0	0	6	33	50	6	0	0.67	0.88	0.05	0.16	0	3000	6
P2	0	0	3	42	47	4	2	0.5	1.25	0.03	0.87	1	3800	4
P3	1	0	4	40	58	4	0	0.54	0.82	0.05	0.1	1	3550	5
P4	3	1	4	22	44	19	2	0.5	1.04	0.04	0.17	1	1100	22
P5	0	0	4	42	48	1	0	0.52	0.4	0.04	0.29	2	4050	1
P6	0	0	1	44	48	4	0	0.48	0.84	0.03	0.16	0	4200	4
P7	0	0	2	46	50	2	0	0.47	0.98	0.03	0.17	3	4300	2
P8	0	0	8	38	49	4	1	0.46	0.59	0.04	0.26	0	3600	4
P9	0	0	4	41	48	3	0	0.52	1.03	0.04	0.73	0	3850	3
P10	0	0	5	40	48	4	0	0.52	0.86	0.04	0.14	0	3850	4
P11	0	0	1	47	48	1	0	0.45	0.26	0.03	0.2	0	4450	1
P12	1	1	9	34	48	6	0	0.46	0.9	0.05	0.12	1	3050	7
P13	0	0	2	45	49	2	0	0.5	0.53	0.05	0.24	3	4300	2
P14	0	0	8	31	49	9	0	0.56	1.13	0.06	0.2	3	2500	10

Table 20. Features from gameplay from the second round of the Set 2.

Participant	No Reaction		Early	On Time		Late		Mean		ST-Dev		Wrong Color	Score	Lives Lost
	LP	NP	LP	LP	NP	LP	NP	LP	NP	LP	NP			
P1	3	3	10	23	52	10	6	0.62	1.05	0.17	0.39	6	1950	11
P2	2	1	6	35	41	5	5	0.53	0.74	0.1	0.61	9	3150	7
P3	4	0	5	23	40	11	5	0.56	1.06	0.06	0.32	6	2100	14
P4	10	7	4	12	26	24	8	0.51	1.23	0.15	0.43	3	0	34
P5	0	2	10	31	46	6	0	0.52	0.55	0.07	0.38	16	2050	6
P6	3	21	15	23	19	8	3	0.49	1.21	0.05	0.34	25	700	11
P7	0	1	8	31	47	11	1	0.49	1	0.07	0.24	6	2500	10
P8	0	0	9	37	47	4	1	0.47	0.75	0.05	0.29	6	3200	4
P9	0	1	6	37	47	5	0	0.52	0.97	0.04	0.7	3	3300	5
P10	1	3	6	33	38	5	3	0.5	1.1	0.09	0.38	9	2400	8
P11	3	2	9	31	47	7	0	0.48	0.98	0.04	0.42	12	2150	10
P12	7	3	4	19	40	19	2	0.5	1.13	0.07	0.38	7	800	25
P13	6	3	5	25	39	13	7	0.56	1.09	0.15	0.58	11	1500	19
P14	4	6	15	22	41	6	0	0.53	1.22	0.07	0.44	10	1500	10

Table 21. Features from gameplay from the third round of the Set 2.

Participant	No Reaction		Early	On Time		Late		Mean		ST-Dev		Wrong Color	Score	Lives Lost
	LP	NP	LP	LP	NP	LP	NP	LP	NP	LP	NP			
P1	0	0	8	33	51	4	0	0.65	0.89	0.05	0.21	1	3100	4
P2	0	0	4	45	50	0	0	0.5	1.4	0.04	0.78	0	4450	1
P3	0	0	7	36	48	5	0	0.55	0.73	0.05	0.2	0	3350	5
P4	1	1	6	24	47	18	0	0.58	0.99	0.4	0.16	2	1450	14
P5	0	0	2	45	49	2	0	0.53	0.46	0.03	0.37	0	4400	2
P6	0	0	4	45	49	0	0	0.47	0.84	0.03	0.15	1	4300	0
P7	0	0	2	46	49	2	0	0.46	0.88	0.03	0.13	1	4400	2
P8	0	0	9	39	49	1	0	0.46	0.49	0.03	0.15	0	4100	1
P9	0	0	2	45	49	2	0	0.52	1.32	0.03	0.69	1	4200	2
P10	0	1	6	40	47	2	0	0.52	0.81	0.04	0.09	1	4000	2
P11	0	0	5	39	49	6	0	0.46	0.5	0.04	0.28	1	3450	6
P12	0	0	13	27	49	10	0	0.47	0.84	0.05	0.17	2	2300	10
P13	0	0	3	44	48	1	0	0.49	0.41	0.02	0.25	0	4450	1
P14	0	1	4	42	47	2	0	0.55	1.04	0.04	0.12	2	3750	3

Table 22. Features from gameplay from the fourth round of the Set 2.

Participant	No Reaction		Early	On Time		Late		Mean		ST-Dev		Wrong Color	Score	Lives Lost
	LP	NP	LP	LP	NP	LP	NP	LP	NP	LP	NP			
P1	1	3	9	23	34	11	5	0.76	1.18	0.22	0.41	6	1600	14
P2	0	0	5	37	43	6	4	0.5	1.01	0.04	0.68	5	2850	5
P3	3	1	3	31	34	11	6	0.58	1.09	0.11	0.37	4	2400	14
P4	10	3	3	21	36	14	9	0.54	1.28	0.09	0.47	4	1000	24
P5	0	2	10	32	44	7	3	0.52	0.7	0.1	0.43	6	2650	7
P6	0	4	8	32	41	9	2	0.47	1.06	0.04	0.29	6	2850	7
P7	0	1	12	25	42	12	6	0.54	1.17	0.34	0.36	14	1850	12
P8	2	0	6	31	45	5	3	0.47	0.72	0.07	0.32	5	3150	6
P9	0	0	9	32	48	8	1	0.52	0.98	0.1	0.74	6	2300	9
P10	1	0	4	38	43	5	2	0.5	1.03	0.12	0.3	3	3350	6
P11	1	2	6	37	48	6	1	0.47	0.68	0.04	0.29	3	3300	6
P12	0	0	8	29	48	11	0	0.53	1.04	0.38	0.17	2	2300	17
P13	0	0	6	37	49	7	0	0.48	0.77	0.04	0.31	3	3200	7
P14	6	5	2	32	39	6	1	0.57	1.11	0.07	0.3	3	2450	14