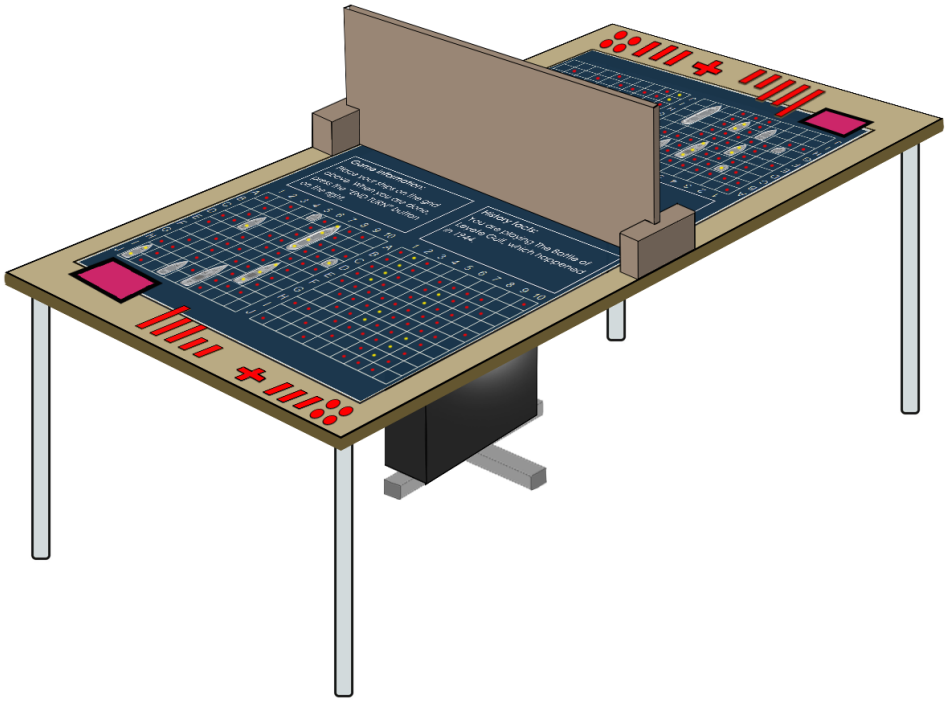
Teaching History Through an Interactive Table

Visual Computing-Human Perception



Project Report
MTA19337

Aalborg University
Electronics and IT

Here you can write something about which tools and software you have used for typesetting the document, running simulations and creating figures. If you do not know what to write, either leave this page blank or have a look at the colophon in some of your books.

Electronics and IT
Aalborg University
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Teaching History Through an Interactive Table

Theme:
Visual Computing-Human Perception

Project Period:
Fall Semester 2019

Project Group:
MTA19337

Participant(s):
Mikkel Sang Mee Baunsgaard
Freja Bøcher Kaastrup Johansen
Andrei-Calin Mares
Dávid Mockovský
Magnus Kornbeck Thomsen
Oscar Bill Zhou

Supervisor(s):
Malte Pedersen

Copies: 1

Page Numbers: 89

Date of Completion:
November 18, 2024

Abstract:

This paper describes the implementation of the Battleship game on an interactive table, where the purpose of the game is to teach pupils history through the gameplay. The interactive table consists of a projector, table with semi-transparent surface and a camera. The report describes the areas considered when creating this implementation. Furthermore it describes the design process including the game design, Low Fidelity prototype testing, and explains the algorithms of the image processing methods used.

Contents

Chapter 1

Introduction

In recent decades games have been used as a tool for entertainment. The hours spend gaming amongst children and teenagers are higher than ever [30]. In the UK, pupils aged 12 to 19 years have increased the amount of time spent playing video games by approximately three hours per week, going from 10.7 in 2013 to 13.8 hours a week in 2018. With the pupils having such a huge time investment and interest in games, it is a shame that the educational institutions do not incorporate games more as a tool to teach and motivate to learn.

Despite games having potential, their educational value is usually downplayed, and games are seen by many people as just entertainment. It is therefore difficult to convince educational institutions to incorporate gaming as a tool to teach. However, if the games were to be modified and be educational, then it might be possible to see an increase in game usage as a means to teach and motivate pupils to learn in the future.

The goal of this project is to create a proof of concept of a game that could be used as a tool to teach and motivate pupils in the age of 12 to 19 years. This is to be achieved using an interactive table and image processing.

Chapter 2

Problem Analysis

2.1 Initial Problem

Many existing games are using history as a part of their "theme" to create stories and scenarios - however, they do not necessarily teach history. Examples such as "Banished" [24] , which is a video game about colonisation and "Battleship" [2], which is a board game about naval warfare, both have the potential to teach history, while in reality, they do not make use of this potential. This leads to the following problem statement:

> "History-based games do not necessarily teach history."

The goal of this project is to expand on this potential and incorporate educational history material into a game which bears the potential to teach it but does not already. This leads to several research questions, among which are questions about how games that are already teaching history do it and questions about which platform (e.g. board game, video game) is preferred. This will be explored further in Section 2.2.

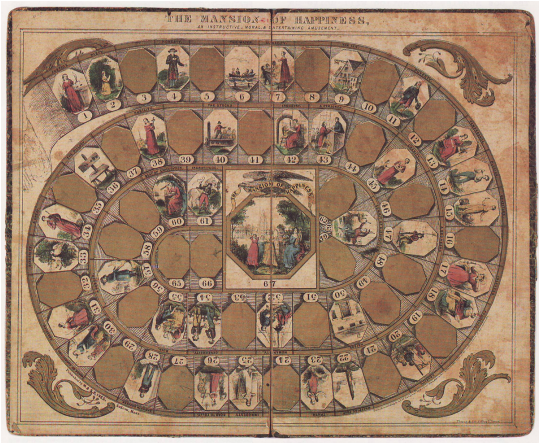
2.2 Background Research

2.2.1 Evolution of Games

Based on Flanagan's book "Critical Play: Radical Game Design" [10], some of the earliest evidence of human play is said to be the board games uncovered in ancient burial grounds or depicted in ancient drawings and carvings. These games have evolved greatly since those times, in terms of the target audience and their purpose. At first, they were mainly popular with lower classes, as a form of entertainment, and were made from simple pieces created with materials such as stone,

wood and/or earth. However, over the years they have evolved, becoming popular with the upper classes, as well as improving their appearance with more valuable materials being used for game pieces. However, not only the aesthetic and popularity of board games has changed over the years. From ritual invoking to assisting in the development of conceptual processes, the purpose of these board games has changed drastically through time.

An example of these drastic changes is the implementation of teaching through certain board games. According to the same source, one early example comes from America in the 1840s.
The game "Mansion of Happiness", as seen on Figure 2.1a, was published with the purpose of teaching children the virtues necessary to live a happy life, while teaching them to avoid sins that can destroy that happiness. This game's purpose was clearly displayed on the box and the creators hoped that children would take these principles to heart, and so, they could reinforce the high moral principles of its time [10]. In the same period, another game with a similar purpose was introduced, "The Checkered Game of Life", as seen on Figure 2.1b, where the players were expected to collect points in order to achieve a variety of habits, which included personal virtues [10].



(a) *"Mansion of Happiness", a board game from 1843. Image source: [16].*

(b) *"Checkered Game of Life", a board game from 1860. Image source: [8].*

Figure 2.1: *Early board games used for teaching.*

Board games have changed and evolved with society, which is why a link can be made between the major world events, the sale and the purpose of board games. For example, during the Great Depression, "Monopoly" (a game about managing your finances) increased in sales tremendously [10]. Likewise, board games keep evolving as modern times do.

Nowadays, board games are extremely varied, and there is much that can be taught using them.

One example is the game "The Catan: Oil Springs Scenario", as seen in Figure 2.2. Some researches [7] call Catan an entertainment game with sustainability concepts embedded in it. They argue that because of its popularity and key sustainability concepts, it is a powerful tool that can be used to introduce these concepts to a wide audience.

Figure 2.2: *Setup of the game "The Catan: Oil Springs Scenario". Image source: [27].*

Another modern example of teaching through board games is a competitive educational guessing game called "Orbital Battleships". The advocates for this game state in an article [15], that the importance of games as a teaching method in a modern world cannot be overestimated. In the same article, it is argued that they can be both fun and useful for any age group, not only kids. Furthermore, it is stated that games are a powerful educational tool that is very flexible. In the case of "Orbital Battleship", the objective is to teach concepts such as the value of quantum numbers or understanding of periodicity. According to them, the game succeeded in teaching the set educational goals.

Similarly, the modernisation of schools and education also brought the idea of teaching via video games. There are arguments that video games can improve teaching because of their entertaining and interactive nature. An article [25] claims that games spark interest in students and aspire creativity and self-driven interest, in contrast to most modern-day schools that present information mainly through

reading, which can be monotonous and disengaging. Creating self-driven interest is important in learning as it can lead to creating communities, which in turn leads into improving social interactions and create collective intelligence.

In order to make the best possible experience for the users, it is important that the games are collaboratively made between developers and educators. The game should be created in a way so it is entertaining, while providing academically accurate information. However, teaching through video games should not completely replace current educational systems. Instead, there is a desire to combine them in order to create the most efficient educational environment [25].

2.2.2 Using Games for Teaching History

With so many precedents where games were used to teach principles or concepts, it can be argued that games could be used to teach players history. This is based on previous statements from researchers about the flexibility of what games can teach and achieve, as well as their target groups flexibility. This section will explore different game types, the potential of games being used for history teaching and existing solutions, which approach that area.

Fundamentally, there are six types of games according to Michela Mortana et al [23]. She describes the common traits that usually find a place within each respective genre. Every genre has its own strengths and weaknesses when it comes to achieving the goal of the game, which acquires different approaches. This process requires the players to adopt new ways of thinking, therefore, the player has the potential to learn along the way. The game types are the following:

- **Action games:** Action games are rarely connected to cognitive gain, as their main mechanics revolve around accurate and quick tasks, such as shooting and avoiding obstacles, which is engaging but not necessarily fit for teaching history [17].
- **Strategy games:** Strategy games are great for teaching tactical and strategic thinking, as the player has to come up with a plan based on e.g. placement of troops and careful thinking to achieve victory. Regarding teaching this genre is good at building this particular skill-set. It is often used for history-based games such as the video games series "Total War" [31], which is set in different time periods and revolves around war and military strategy [17], with real time stamps and names, which are historically accurate to some extend.
- **Simulation games:** These games usually simulate certain situations that make the player step into the shoes of another person, either based in fic-

tion or reality. These games are good at teaching the player about the empathy of the people and circumstances they are in, varied from game to game. They have the potential to teach awareness and moral-related issues based on historical events [17].

- **Trivia games:** Trivia games are usually built as question-based games, where the players gain points or some sort of progress the more answers they get correct. The categories or topics of these games are near limitless, which includes history [17].
- **Puzzle games:** Puzzle games are usually relatively short games, that revolve around using logic or navigation to solve a puzzle, which is good for training critical and logical thinking, but could be difficult to teach history with [17].
- **Adventure games:** Adventure games is a broad category, however, they usually rely on the player to find information and applying this newly acquired knowledge to fulfil a task to progress. This approach is "learning by doing" where the player actively has to seek out information. This approach is related to the Constructivism in education theory [17], and makes the player more likely to remember the information as it is not provided passively by the game [17].

Taking these genres into consideration, the following board games can be mentioned in regards to teaching history.

Axis and Allies

Axis and Allies [4] from 1981 is a cooperative team game revolving around World War two on a fully global level with a map divided into chunks like the game RISK [4]. A part of this map can be seen in Figure 2.3a along with the pieces and resources being administered. Here the players are playing as either The Axis or The Allies, where the goal is to capture as many major cities as possible, which they do by coordinating offences and utilising their production points. By focusing on the global scale of World War two it gives players an insight into how hard it was to organise both military and wartime economy during the war, as well as logistics, regions and general knowledge about the war. Due to the fact that players are in control of how the war progresses and ends it is, however, important to know that the game does not depict accurate history.

The Grizzled

The Grizzled [4] from 2015 is also a cooperative board game, however, it has more emphasis on the more gritty aspect of war. The setup of the game can be seen in Figure 2.3b. The players get in the skin of individual soldiers where they have

(a) *"Axis and Allies" board game. Image source: [1]*

(b) *"The Grizzled" board game. Image source: [28]*

Figure 2.3: *Board Games that are teaching history.*

to survive together as a squad until the armistice. A squad consists of one team leader and normal infantry soldiers, each player has different traits they have to play accordingly in order to win the game. If one team member dies the game is lost, which makes the focal point revolve around planning and teamwork. Compared to Axis and Allies players gain an insight into the hardships and pitfalls of the individual soldiers, which brings another perspective to wars. Instead of having a very superficial approach to a warlike Axis and Allies, it instead brings a more harsh approach to it. The players will learn the realistic side of battles and therefore gain a better understanding and emphasis of the conditions of the soldiers during the war.

In conclusion, there already exist board games, which can teach certain aspects of history. However, the actual learning outcome differs depending on the genre and the context of where the games are played. Depending on the context people may play the game either at home, at a friends place or a public place. If the game is played either at home or at a friends place the players have more time to properly immerse themselves into the setting and therefore learn from it. On the contrary, if it is played at a public place the immersion is limited due to the circumstances that can affect the gameplay [17]. The example solutions mentioned previously both revolved around the strategy genre due to the cooperative and turn-based approach. Furthermore, it is a genre that is easy to adapt into a historical setting which is why it is one of the most common genres for history-based games. Taking this into consideration makes it the most appealing genre to use as a teaching tool for future history games.

2.2.3 Methods of History Teaching Through Games

As mentioned before, using games as a medium for teaching is progressively getting more and more popular [9, 26]. Especially video games are popular among adolescents and young adults, which according to Richard E. Clark is arguably one of the most troublesome age groups for educators [9]. Additionally, Prensky et al. believes that the learning aspect from video- and computer games is what is attracting kids to them [21].

Focusing on history, it is necessary to know *how* history is taught and additionally how it is taught through games. Schrier et al. describes three methods used for educating history in the book "Learning, Education and Games" [23]. These three types are called "Best possible story", "disciplinary history" and "postmodern history". In the first type, which will be the type used throughout this report, the aim is to teach "the most agreed upon" version of history. However, this can prove to be difficult, since there is a lack of agreement around how history progressed. The second type aims to let the students "weigh perspectives of the past" and the third type aims to questions if it is possible for historians to construct history without being affected by their own biases and choices.

Moving on to history games, Schrier et al. mentions three types of history games. The three types are "Representation of the past", "Interaction with historic themes, concepts, choices and resources" and "Play within a historical or history-related setting". In the first type (representation of the past), the goal is to recreate a specific moment of history, which is then re-performed by the players. In these games, the creators make great efforts to make the moment as accurate and attentive to historical detail as possible. The second type (Interaction with historic themes, concepts, choices and resources) is more of an opposite from the first, since these games are generally less focused on historical accuracy. Here, the players are making a decision and using resources, more or less acting as a "godlike player" [23]. This type of historic game is more focused on letting the players consider "relevant historic questions, causes and effects and/or systematic issues" [23]. "Axis and Allies" and "The Grizzled", which were mentioned in section 2.2.2, both fall under this type of historic game. The third type (play within a historical setting) can have features of both the first and second type. It is not as focused on historical accuracy or using specific historical moments - however, it is very focused on accuracy concerning the cities or areas the games take place in. The games turn into "historical artefacts", since the attention to detail is so great, that the player gets to experience the areas, as they were in that time frame.

2.2.4 Combining the Physical and Digital World

As argued in section 2.2.1, video games have been successfully used in order to teach the players, and board games have the potential to be used for the same purpose. The same methods that video games use could be integrated into a digitised version of a board game in order to facilitate the teaching of history. This integration would be able to take advantage of the features of both mediums, but the combination has to be done in such a way as to not lose the social aspect of a board game. The reasoning behind that conclusion will be further explored in this section.

The first area that needs to be explored for this is if board games need to be digitised and if so, to what extent. For this purpose it is needed to look into the advantages and disadvantages of video games in comparison to board games and what kind of a mix between these two already exists. Video games have the computational advantage where they can create virtual environments, control in-game artificial intelligence units or keep constant track of the state of the game. On the other hand, they take a lot from the social aspect, in which board games excel. Social interactions in video games can differ a lot from physical interactions. As described in a paper by José P. Zagal et al. [33] players online tend to be very hostile or "toxic" instead of cooperative or competitive in a healthy way, which can ruin the game for other players. It also claims that players tend to behave competitively in digital games even when playing cooperative or collaborative games. This is a behaviour that can be more easily avoided through physical contact and is generally not present in board games. However, the computational advantages can still be implemented in a board game, therefore making it easier and more interesting to play. One study [13] made an experiment in which they tested if people prefer traditional board games, their digital version or a mix between the two. In the mix of the two, some aspects were handled by a computer and projected on a table, while keeping the physical interaction of players with each other and the game itself. The results of this test showed that people preferred the combined interaction-display format from both the social aspect and personal game experience point of view. To achieve this outcome it is required to find a balance between making a board game to a digital version while keeping enough physical aspects. This is also known as making a pervasive game, which will be explored in Section 2.2.5.

2.2.5 Pervasive Gaming

Pervasive gaming is a way of extending the virtual gaming experience into the real world by connecting game elements tied to a virtual world with traditional game elements tied to the real world. The basic concept of pervasive gaming is that players in the real world use a computing device to get certain information tracked, e.g. context or location [3]. The concept of pervasive gaming has been used before to create tabletop games, such as the STARS platform for an augmented table [3]. In this section about pervasive gaming, the focus will be on these augmented table games (interactive tables).

While most pervasive games attempt to transform the real world into a game board, augmented tabletop games take the opposite approach by building directly on the success of old-fashioned board games. These products take advantage of their strong social situations, while enriching them with digital benefits to realise a new hybrid form of entertainment. In other words, this combination has the potential to combine the best of both worlds by favouring direct interaction between human players while a computer keeps track of the game state. The computer is additionally able to provide atmospheric visual- and/or audio assets, as well as taking the role as a participant [3].

One example when this form of pervasive gaming was used successfully was the STARS platform [3]. This example "uses wireless communications to integrate sensing devices with a smart game table, providing various means to bridge the gap between the real world and the virtual world, supported by a software platform that facilitates prototyping of tabletop games" [3].

Figure 2.4: *The KnightMage game - a game where the physical board is replaced with a digital screen, but the game pieces (e.g. dice, minions) are physical. Image source: [14].*

One of the STARS platform games is "KnightMage". As seen in Figure 2.4,

this game is a combination of a digital and traditional board game. It is a medieval hack-and-slash type role-playing game where players cooperate to explore dungeons, landscapes and fight monsters, while competing against each other for treasures. In this game, great care is taken regarding which game elements are represented in software or are left to the group in the real world. For instance, the rolling of dice remains an engaging physical activity. In these ways, KnightMage aims to combine the emotionally involving social situations of tabletop role-playing games with the typical features found in corresponding computer adaptations [3]. This is a principle that will also be followed for the solution proposed in this report.

2.2.6 Target Group

As stated in the problem formulation, in Section 2.1, the focus of this project is teaching history. Therefore, a potential target group for this project are pupils that have history as a subject in the curriculum. This category incorporates children/teenagers from age 9–19 from both primary and secondary schools as well as high school. Specifically in Denmark history is taught in primary schools from 3rd–9th grade and throughout the entire high school at A level [29]. However, in this project, the focus is put on the age of 12–19. The reason behind this is that from the 6th grade, the amount of history lessons is doubled [29]. This can be a period in which pupils need more help, as the workload drastically increases. Another reason is the fact that the interactive table might contain graphical interface of battles, which might be considered as a depiction of violence. Therefore a PEGI (Pan European game information) [32] age labelling is taken into consideration, where the age of 12+ is said to be permitted to display violence in a slightly graphic nature.

Furthermore, the early adolescent brain (starting from 10 years old) is going through a growth spurt just before reaching puberty. In this spurt, the brain goes through a so-called “pruning” phase where the parts of the brain that are used the most are strengthened and those that are unused will deteriorate. This process will continue throughout adolescence and will, therefore, strengthen the intellectual activities. This period will influence the adolescent as a learner for the rest of his/her life [22]. So, it can be said that during this period, adolescents have a big potential for learning new skills and knowledge.

As has been previously mentioned in Section 2.2.3, adolescents are one of the most troublesome age groups for educators, when it comes to traditional teaching. The reason for this, as Prensky et al. [21] argues, is motivation. Unfortunately, much of the content that needs to be learned is considered to be “boring” or “dry” by students. Same researches argue that teachers, trainers, and educators are rarely as effective as they might be in the motivational department, and this often causes

problems in getting students to learn [21]. These researchers consider game-based learning to be a solution to this lack of motivation. It can be argued that games are highly motivating "vehicles" [9] and provide learning opportunities at every second [21]. However, game-based learning is not widespread. Clark et al. [9] argues, that the main reason for this is the fact that educational-decision makers may, out of bias or lack of understanding, discount or discourage the investment in games, and so ignore an innovative way of teaching.

To conclude, based on the previous arguments, since pupils are actively learning history in school, but don't consider the traditional teaching way motivational, they seem to be an appropriate target group for an interactive game that teaches history. This, plus the fact the area of game-based learning hasn't been fully explored yet, can help conclude that an appropriate target group for this project is pupils between the age of 12–19 years.

2.3 Computer Vision

As the implementation is set around an interactive table, it is important to explain what aspects are relevant for it to work properly. The first aspect explained will be computer vision, as it is the main input in this case. Computer vision is a term used to describe the advanced algorithms similar to the ones humans can perform [18]. Humans are capable of extracting information far beyond what is written in text, hence referring to *"a picture is worth more than a thousand words"*. This is what computer vision tries to replicate, with the eyes being replaced by a camera and the brain is replaced by a processing system [18]. In this project, the open-source computer vision and machine learning software library OpenCV [19] is used. Furthermore, the following content will, to the greatest extent, be based on the book by Thomas B. Moeslund [18]

Chapter 3

Initial Design

Chapter 2, in which the initial problem is analysed, helps point to an objective for this project. This objective is to create a system that can digitise a board game with the aim of teaching pupils history.

As discussed previously in Section 2.2.4, people prefer the combined interaction display format rather than a traditional board game or a video game. A balance between the computational advantages of a video game and the physical aspect and social interaction of a board game has to be struck. To achieve this, an interactive table, such as the STARS platform discussed in Section 2.2.5, is a desirable solution.

Working with an interactive table raises questions about what components come together in order to create such a system, as well as how the system should be set up. These questions will be explored further in this chapter.

3.1 System Components

An interactive table system usually contains three main elements that need consideration: a *display device*, a *table-surface* which works with the display device and a *sensory device*.

3.1.1 Displaying Device

For the interactive table, a displaying device is required to create an image. Possible options for this are screens or projectors. Screens are of many types e.g., as normal LED screens, touch screens and flexible screens. While a screen can be used to easily display images in high resolution, they are usually not accessible to students at educational institutions, especially for the purpose of being integrated into a table. Even at home, this can pose problems as a new screen has to be bought for the special purpose of being part of the system. The same goes for projectors,

which are rather expensive - however, many teaching institutions provide projectors which can be used.

For the reasons stated above, the display device chosen for this system is a projector. This means the projection placement needs to be considered in regards to the surface it is projected onto.

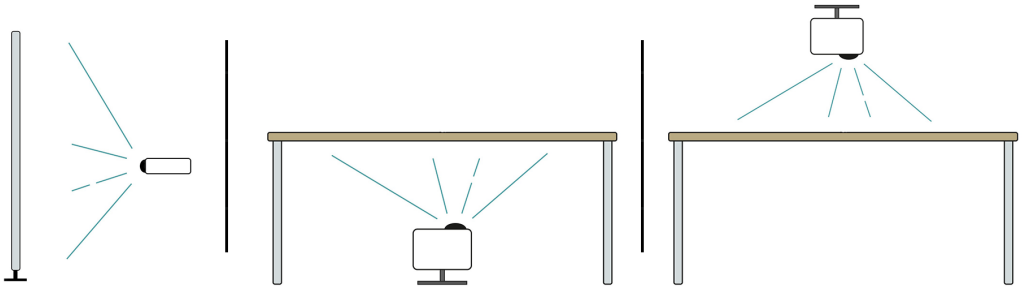


Figure 3.1: *Illustrating the three different setup options mentioned in 3.1.1 - shown in the order vertical projection, bottom-up projection, top-down projection.*

- **Vertical projection** - this setup requires a wall or a whiteboard. The entire setup would be angled by 90 degrees relative to the floor and the projector would project sideways, as seen the most commonly. The camera would be placed next to the projector. This setup would require a way to keep the physical objects stick to the wall, such as magnets, while making it possible to move them with ease. A possible problem would be setting the projector up in a way, where it does not interfere with the users moving in front of the board.

- **Bottom-up projection** - this method requires a table surface that will allow the light from a projector to pass through the table. The ideal situation would be to let enough light pass through, so the image is sharp, while not blinding the user. This setup also requires a special wide-angled projector, as the table can't be too high for convenient use. The camera placement could be either next to the projector or above the table. Having the camera below the table could make detecting objects placed on the table difficult. The camera placed above the table, however, could potentially create the same problem as the previous setup.

- **Top-down projection** - where the projector is above the table, projecting downwards on the table. It is accompanied by a camera that is right next to the projector to have the same viewing angle on the table as the projector. This setup needs some sort of ceiling mount for the projector and the

camera, or a construction that can support the weight of the projector while being safe and high enough. It is important to keep in mind, that the player's movements above the table would create shadows over the projection.

For the purposes of this prototype, the bottom-up projection will be used. This will generally help with a simpler setup compared to the top-down projection, which would require either a ceiling mount or a construction to hold the projector. Furthermore, both top-down and vertical projection would be significantly affected by shadows, whenever a player moved in front of the projector. This means that the prototype requires a table setup with a surface, which light can pass through.

The chosen projector for the prototype is a short-throw projector model: InFocus IN1245Ta, which means that the distance the projector needs is much smaller than a standard projector. This allows a bottom-up projection, without having to raise the table to an uncomfortable level as mentioned before. In this case, the projector has to be set up at a distance of 62 cm from the lens to the table. In this case, it creates an image that is 100cm long and 62cm wide on the semi-transparent sheet of the table, thus covering the majority of it.

3.1.2 Surface

Secondly, the nature of the table surface itself should be discussed. With that in mind, it can be either a semi-transparent table or an opaque table which each having its advantages and disadvantages:

- **Semi-transparent:** This surface has the advantage of the table being translucent, which means that a projector can be placed beneath the table. This setup has the advantage of projecting without blinding the user, while maintaining the sharpness of the projection. However, as the projection is only happening beneath the table, it would only project on the table itself, and not project something on top of an object placed on the table, unless that object is also to some degree translucent.

- **Opaque:** The opaque table means that there has to be a "top mount" or ceiling mount, for the projector to project downwards. However, the distance between table and projector should be taken into consideration, as well as how much space it would take up before it becomes too impractical for certain environments, where the table might be placed. Since the projection is happening from above, both the table surface and the objects placed on the said surface are able to have something projected on them. However, if the objects are three-dimensional, shadows occur, which can cause some difficulties.

The table chosen is a semi-transparent one, since this one allows for more flexibility in the possible projection setups, allowing for images to be projected from below. The table used for the prototype is made of wood and has the following dimensions: 155 cm in length and 75 cm in width and stands 93 cm tall. It has cut out space for a semi-transparent plastic panel in the middle for projection and the dimensions of the screen are 125 cm in length and 69 cm in width. The height or thickness of the panel is under half a centimetre.

3.1.3 Sensory Device

Other than the surface of the table and a display mechanism, the sensory device used to capture information about what is happening on the table should also be taken into consideration. This sensory device can capture the information in many ways such as audio, video and proximity.

However, the device that would make the most sense, in this case, is one that can capture video information. In this way, the placement of possible pieces and the actions that the players have to make can be easily captured. In opposition, the other options do not provide enough information in order to create an accurate representation of these things.

Based on this, the sensory device chosen is a camera. It is the optimal choice, since it registers a live feed of the game board. Then, through image processing, it can capture colour and shapes. The camera chosen is a webcam, and the model is Logitech C270. The possible placements for the camera are identical to the possible placements of the projector: vertical-, bottom-up-, and top-down placement. However, they do not necessarily have to match the position of the projector. For the purpose of this prototype, the camera will have a top-down placement. This allows *one* camera to capture the entire table, giving it the opportunity to clearly perceive colours of any physical pieces on top of the table. For the particular camera used for the setup, the distance from the lens to the table has to be around 118 cm.

3.1.4 System Setup

By combining the decisions that were made in Section 3.1.2 and 3.1.3, it is possible to create a setup for the system to run work properly. The displaying device, surface and sensory device are meant to be static throughout the game. The whole setup can be seen in Figure 3.2.

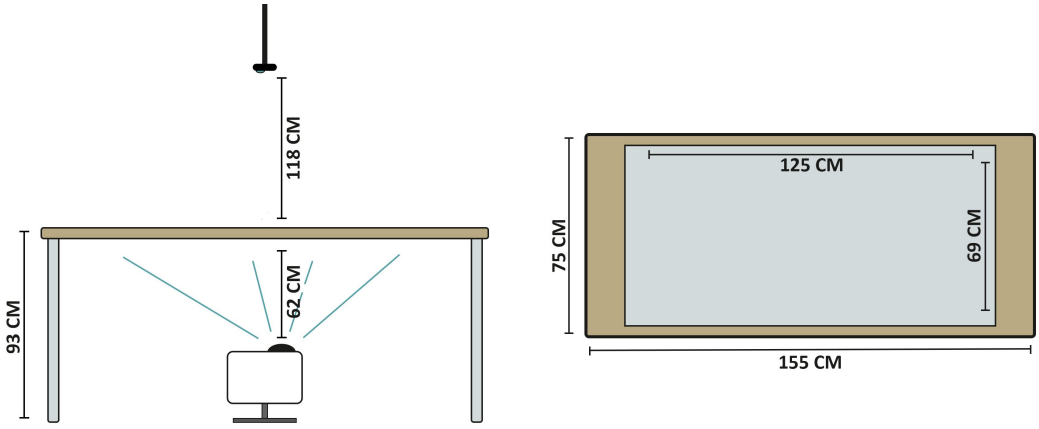


Figure 3.2: *Side view and top-down view of the setup, one showing the measurements between the displaying- and sensory device, the other showing the dimensions of the table and the surface*

Furthermore, a simplified diagram showing how the system components interact with each other can be seen in Figure 3.3.

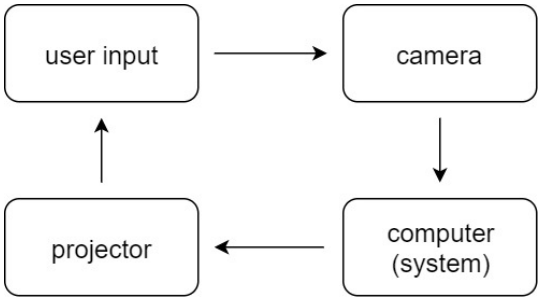


Figure 3.3: *Diagram showing how the system components (and the user) interacts with each other.*

3.2 Choice of Game

Keeping the initial problem formulation in mind, the board game which will be digitised had to be chosen. As a proof of concept, the game "Battleship" is chosen, as it is a history-inspired board game, that does not teach history. The choice is made for several reasons:

- The Battleship game has both physical and digital versions. This makes the process of creating a hybrid version of the two easier.
- The game itself is based on World War 1, however, it does not teach history based on that or any other historical event.

- Since the game is turn-based, it leaves enough time to teach history in between the turns of each player.
- The game has simple rules and is easily understood, making it suitable for pupils.

By choosing Battleship, an additional system component is needed. Since the setup uses a flat surface, it is necessary to consider how to limit the view of a player, to only being able to see their own side of the grid and not the grids of their opponent. The material for this needed to be something that could have a stable upright positioning in the middle of the table, while providing easy setup assembly. The choice for this is a thin (0.5 cm) wooden wall with a 35 cm height and a 71 cm width, which proved to be sufficient in obstructing the view.

3.3 Initial Gameplay Design

In this Section, the gameplay of the original Battleship game is described, and problems that arise when digitising this gameplay are analysed.

3.3.1 Original Battleship Gameplay

The original board game is played on four main battle grids, with each player having two. Traditionally, the players place their own game pieces, which are shaped like ships, on the horizontal grid and attack the vertical grid. Pins are used to indicate miss (white) or hit (red), as seen in Figure 3.4. The grids have slots, in which these game pieces must be placed into.

Figure 3.4: *Hasbro's physical version of Battleship. Image source: [12].*

The number of pins is usually sufficient to fully cover both the grids. The number of ships and their shape varies depending on the version, where the Hasbro

version [2], as an example, contains five ships per player and the grids have a size of 10x10. The ships are classified as follows:

- One “Carrier ship”, which takes up to five spaces.
- One “Battleship”, which takes up to four spaces.
- One “Cruiser”, which takes up to three spaces.
- One “Submarine”, which takes up to three spaces
- One “Destroyer”, which takes up to two spaces

The two grids are separated by an “attack-grid”, a sort of dividing wall, to block the view of the opposite player. Here the players can place the pins to keep track of whether or not they hit the opponents ships. In order to attack the opponent, a player has to audibly announce which cell on the grid they are targeting. They do so by announcing a combination of a number and a letter (e.g. B5). When a ship is hit by the opponent, the player has to announce the hit. The game ends when all the ships of a player have been destroyed.

These gameplay mechanics lead to problems when trying to digitise the board game. Firstly, the game pieces and how they will be placed on the board have to be considered. Secondly, the mechanics of targeting ships and turn switching have to be thought through.

3.3.2 Modifications to the Original Gameplay

Digitising the game creates opportunities for various modifications. These changes can be visual, as well as changes in the actual gameplay. An example of the visual changes can be as seen on Figure 3.5, with the grids placed side by side.

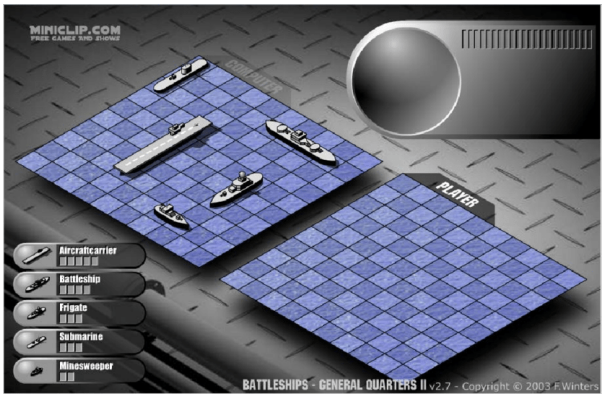


Figure 3.5: *One of many digital versions of the game. Image source: [11]*

Changing the gameplay can be done through the addition of content to the game, e.g. through power-ups. Having power-ups adds many possibilities to new game content, however, since Battleship is about naval combat, they should be restricted within this area. Ideas considered for this prototype are:

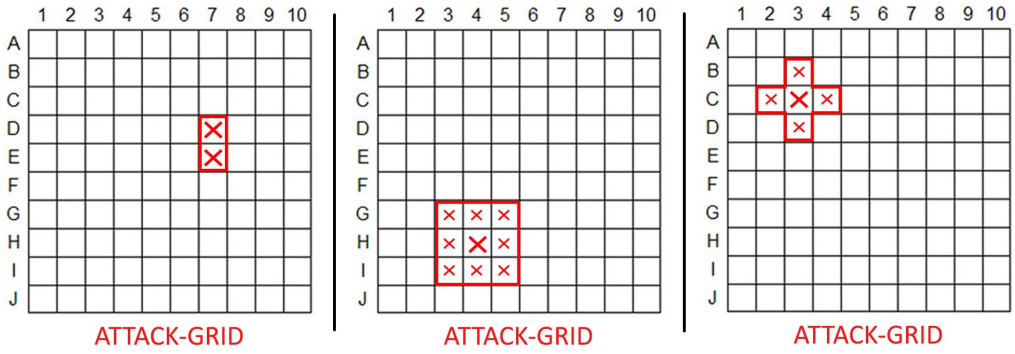


Figure 3.6: *Examples of a player using power-ups. The leftmost image illustrates the "Torpedo" power-up, the middle one shows the "Radar" power-up and the rightmost image illustrates the "Barrage" power-up.*

- **Torpedoes:** A player has a total of three torpedoes throughout a game. The torpedo replaces a regular attack, however, it hits two tiles adjacent to each other. This is illustrated in Figure 3.6.
- **Radar:** Single-use power-up that a player uses instead of shooting. This reveals an area of 3x3, with the attack piece in the centre. This is illustrated in Figure 3.6.
- **Barrage:** Single-use power-up which allows the player to hit 5 tiles, in a cross-shape, in one turn. This is illustrated in Figure 3.6.

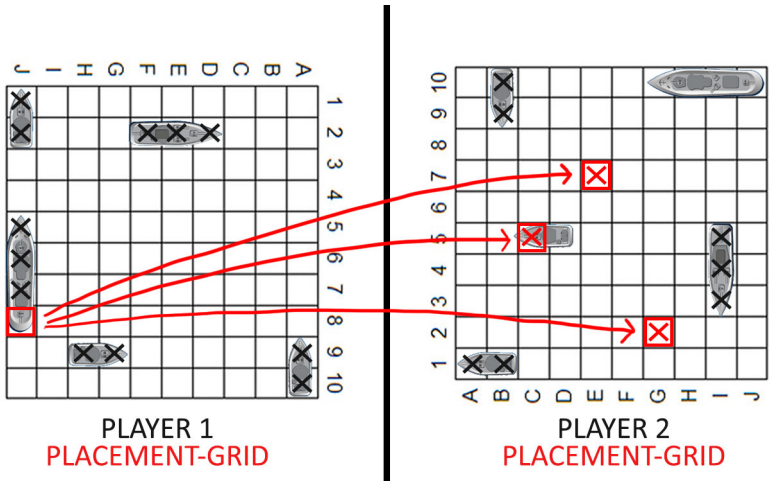


Figure 3.7: *An example of a player using the "Last Stand" power-up.*

- **Last Stand:** When there is only one ship left and the ship ONLY has one tile left it will shoot three times randomly at the opponent's grid. This is illustrated in Figure 3.7.

- **Aircraft Carrier:** A player can use one turn to deploy aircrafts. At the start of that player's next three turns, an aircraft will randomly attack a single tile on the opponent's grid. This power-up will have a cool-down effect of six turns, from the moment it is used i.e. this power-up cannot be activated again during this duration. Additionally, this power-up connected to the players biggest ship (e.g. the Carrier Ship in Hasbros version), meaning that once that ship is destroyed, this power-up will no longer be available.

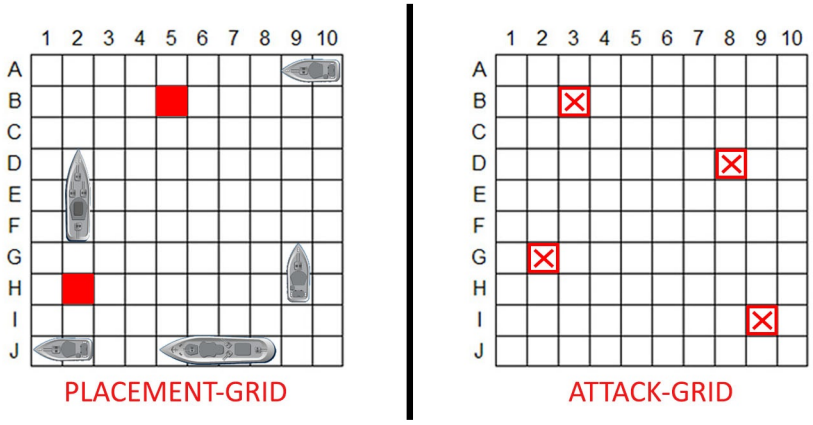


Figure 3.8: *An example of a player using the "Mines/Decoys" power-up. The "Placement-grid", on the left, illustrates a player using the decoys and the "Attack-grid", on the right, illustrates a player using the mines.*

- **Sea Mines/Decoys:** This is a choice each player will make at the beginning of the game. As the players are placing their ships they can choose between placing four 1x1 mines or two 1x1 decoys. Should the player choose the four mines, they would place these mines as desired on their attack-grid. As the game starts, these mines would hit any ships that have been placed on those tiles, in the opponent's grid. Otherwise, the player could otherwise choose the decoys, which would function as a regular 1x1 ship. The opponent would, however, not need to hit these decoys in order to win, meaning that they are only there to confuse them and waste their turns. This power-up would completely replace the 1x1 ships in the original game. This is illustrated in Figure 3.8.

3.3.3 Game Pieces

As mentioned in Section 3.3, an aspect of this project focuses on combining the physical and digital aspects of the game Battleship, and so, it is necessary to modify the traditional game pieces to fit this approach. Based on the setup of this system, an advantage is that historically accurate ships can be projected on the board, instead of using the standard pieces from the tabletop versions of the game. Some materials considered for the game pieces were:

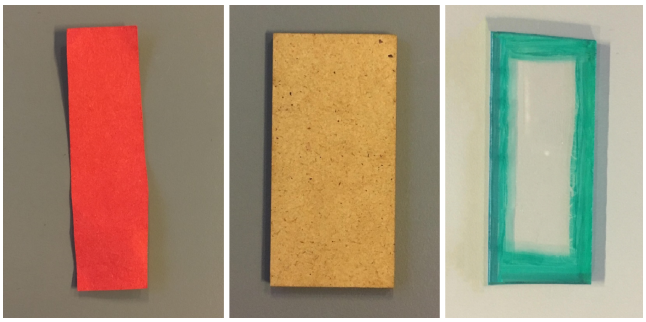


Figure 3.9: *Illustrates the three different game piece options mentioned in 3.3.3.*

- **Paper:** This material is light and thin, which presents both advantages and disadvantages. Although difficult, one can project onto this material. The diverse colours of this material facilitate the use of colour detection algorithms. However, the light weight of the paper poses a big disadvantage. Paper is prone to be moved out of place without the intention to do so (e.g. by lightly blowing at them, they can be scattered), while it is also more easily destroyed (e.g. ripping or crumbling).
- **Wood:** This material can be either light or heavy, depending on the design and type of wood. Additionally, this material is solid, which is an advantage in terms of the pieces staying in place and for maintenance of the pieces. However, since it is completely solid, projection on these pieces, from below, is not possible. This could be solved by carving the actual ship shapes in it. The ships could also be placed onto the table first, at which point their position are registered. Afterwards, the pieces can be removed and a projection of the ships can replace them.
- **Plexiglas:** This material is light and translucent. That makes the projection of ships easy, as light passes through the material. However, it means that the material is also very hard to detect by the camera. This could be solved by painting an outline on the transparent pieces.

The concept of removing the ships after they have been placed could be used no matter the material. For the purpose of the prototype, the wooden pieces were

chosen. This was due to the above-mentioned advantages, and for visibility to the camera.

3.3.4 Attacking and Ending Turns

As mentioned at the beginning of Section 3.3.1, the attacks and the passing of turns in the board game version is done by verbally announcing the moves. This approach can be and have already been improved through the use of the computer. One way that an attack can be performed is explored in this section, and options about ending turns are analysed.

Attacking

Also stated in Section 3.3.1, the original gameplay features an attack-grid placed vertically for attacking. Since the game is digitised, it is more efficient for this attack-grid to be placed next to the player's own placement-grid. This makes both the projection and the visibility for the camera much easier. An illustration of this can be seen in Figure 3.10. This is similar to the existing digitised version in Figure 3.5 if the grid was to be perpendicular to the table, additional equipment would be needed, making the prototype more complex than necessary.

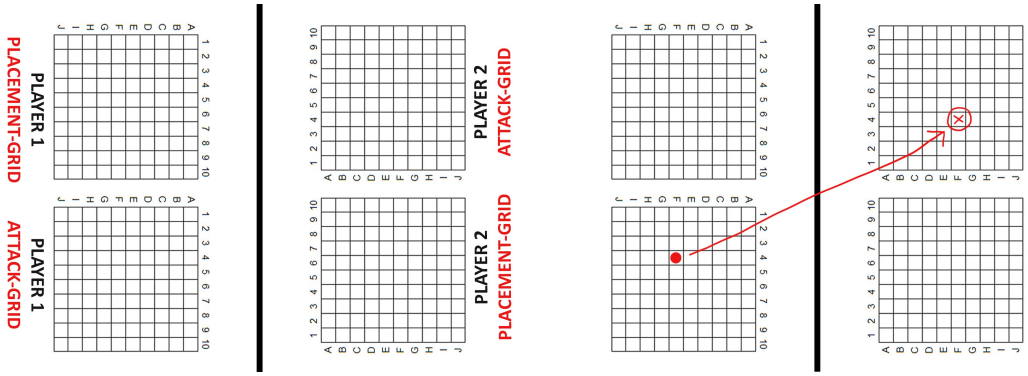


Figure 3.10: *Two illustrations, one showing the player's grids side by side, the other showing an example of Player 1 attacking Player 2.*

In order to attack, a player places the piece fitting the attack of choice on the coordinate they desire. For a regular attack that would be a 1x1 circle piece. This can be seen in Figure 3.10. This circle would be detected by the computer using the input from the camera. The corresponding coordinate on the opponent's board would then be attacked and if it is a hit, that coordinate would change to the colour red. If it is a miss the coordinate would change to blue. The same or the opposite player will then take the next turn, depending on whether the player hit or missed during the first turn. A diagram of this process can be seen in Figure 3.11.

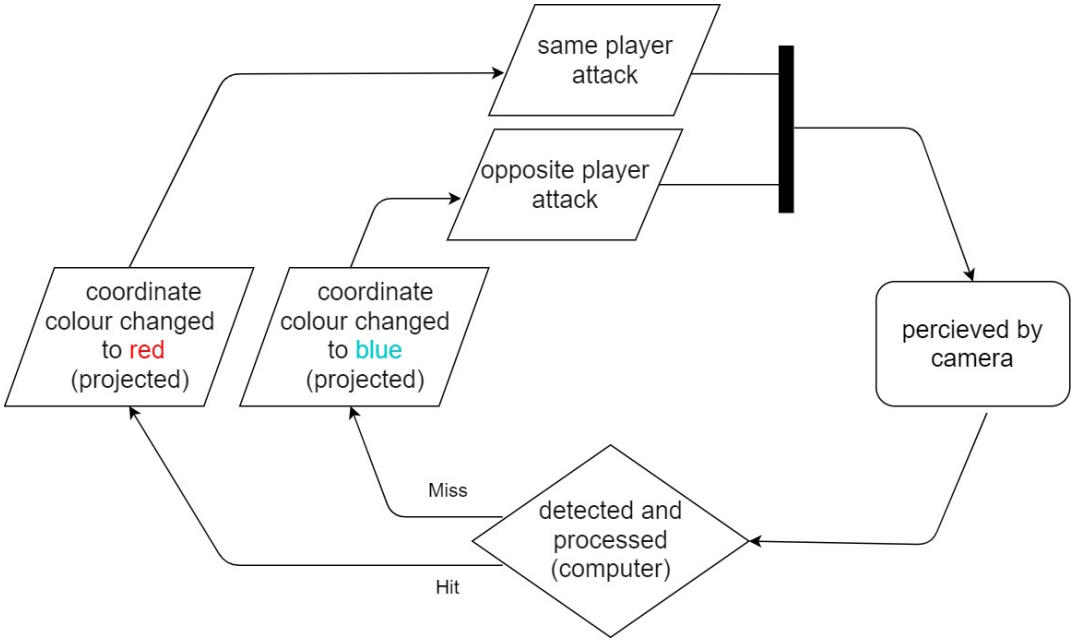


Figure 3.11: *Illustration.*

Ending Turns

To indicate that a player is done with their turn, a confirmation/end turn action from the player is needed. There are multiple options that were considered. To mention a few, a physical button outside the projection area or another game piece that would be placed in a "confirmation zone" were some of the options. Another option, which was chosen for the final version, is having the player do a hand gesture as confirmation in a pink square in the cameras field of view. The image processing software would then recognises the specific gesture and end the players turn. However, the physical button was still used throughout Lo-Fi testing of the system. Since the players will be moving their hands above the table throughout the game, it was decided that this gesture should be done in a specific area of the table. This was decided in order to minimise the case of a player accidentally ending their turn.

3.3.5 Integration of History

Another aspect that should be explored is how history can be integrated into the traditional Battleship gameplay. As mentioned previously in Section 2.2.3, there are three recognised ways of teaching history in games. The most useful for this project is the second one, namely "Interaction with historic themes, concepts, choices and resources". The players will compete against each other in representations of historical battles, additionally being informed of the type of ships they have at their

disposal. Furthermore, the players will have historical facts about the battle presented to them in an information box throughout their gameplay.

The game could contain naval battles such as:

- Battle of Manila Bay (1898)
- Battle of Jutland (1916)
- Battle of Cape Matapan (1941)
- Battle of Leyte Gulf (1944)

It could contain naval battles from different time periods. However, for the demonstration purpose of the prototype, there will be one playable scenario. The battle of choice is the Battle of Leyte Gulf. This is considered to be the biggest naval battle in history, making it a significant event in naval combat to teach about. This battle took place in October 1944 and was a confrontation between the joined forces of the USA and Australia, and Japan. This would allow the players to interact with the historical setting of the Second World War. They would control ships from the period and should be exposed to the tactics and resources available, while maintaining the gameplay aspects mentioned in Section 3.3.2.

To accomplish this, a combination of facts and a more engaging way of teaching will be used. Besides being informed on the real outcome of the battle and the parties involved in the conflict, the power-ups also add a new aspect to the game. It gives the ships more powerful abilities, which would be based on the resources available for the specific battle. This would, as an example, mean that the Aircraft Carrier power-up mentioned in Section 3.3.2 should only be available for naval battles that used aircrafts. Should aircrafts not be an available resource, another power-up would replace it with something more historically fitting for that battle. In this way, players would get to interact with multiple resources, concepts and learn about the evolution of naval combat. This could also ensure that different game modes (choice of battle) are equally interesting, as well as prompt users to try different battles. By using different time periods it is also possible to open up for a wider choice of power-ups - e.g. using canons, instead of torpedoes, in a naval battle with pirates.

In more advanced iterations of the system, more historical periods should be explored. The players would then be exposed not only to the World War two naval setting, but many more. That being said, the first way of teaching, as described in Section 2.2.3, will not be used, since this method implies recreating the past. In this case, the players will be creating their own outcome of the battle, meaning it will become historically incorrect. The players will, however, be informed of the actual outcome of the battle.

3.3.6 Guiding the Players

Since the gameplay of the original Battleship game will be modified, players would have to be informed on these changes in this new game in order for them to be able to play it properly.

This information can be presented through the means of another text box. Here instructions about the game could be displayed. Such as how the attack piece looks like and the steps of how the game should be set up. This will be especially useful, since the players have to remove their pieces for the ships to be projected, as explained in Section 3.3.3. Any other game-related tips would be described here, leaving the historical facts separate as to not distract from them. This, as well as the information box for historical facts, can be seen in Figure 3.13.

3.4 Low Fidelity Prototype

A Low Fidelity (Lo-Fi) prototype is a very basic version of the design concept, which main purpose is to test functionality and usability. Keeping this in mind, the design choices described in Section 3.3, as well as the set up described in Section 3.1.4 can be explored by using a Lo-Fi prototype, without focusing too much on the visual appearance, which is not as important in this type of prototype [6]. In this section, the methodology used and the Lo-Fi prototype of the system is described. The prototype was tested through two tests - the first focusing on the basic gameplay and the second focusing on testing the added power-ups.

3.4.1 Methodology

All methods used throughout the initial design and testing of the Lo-fi prototype are derived from the book Sketching user experiences [6].

The Interaction Design Cycle

The interaction design cycle is an iterative process used to create and test a user experience design. This process is illustrated in Figure 3.12. The cycle consists of four steps and are often revised when gone through based on the results of the testing.

1. **Establishing requirements:** This is the first step of the iterative process. Here needs are identified/refined and requirements are established for the user experience
2. **Designing alternatives:** This is the second step of the iterative process. Here alternative designs are created to meet the established requirements from the first step.

3. **Prototyping:** This is the third step of the iterative process. Here a prototype is created based on the established requirements and designs to allow design ideas to be communicated to stakeholders and assessed by users.

4. **Evaluation:** This is the fourth and last step of the iterative process. Here everything that has been built throughout the process is evaluated as well as the user experience that it offers.

These individual parts of the process are repeated until the evaluation provides a satisfying result, which will conclude as the final product [20].

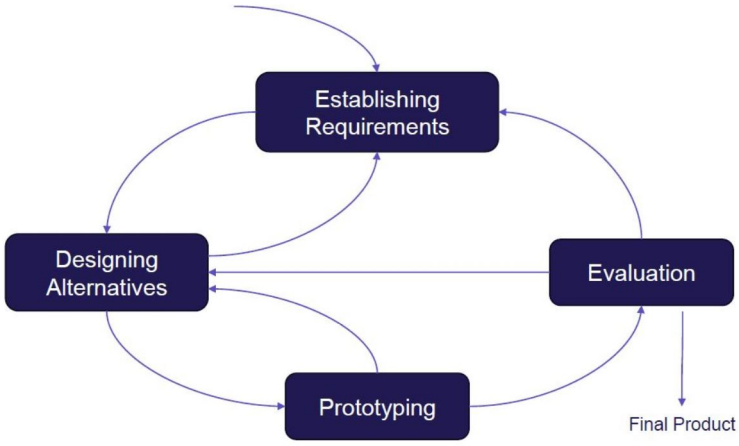


Figure 3.12: *Interaction Design Cycle [20]*

Wizard of Oz

Wizard of Oz is a method used to initiate a system to a certain extent, most commonly for usability interfaces. Here a "wizard" will control the back-end of the system while the user is interacting with the solution. The wizard's job is to react to the user's choices and respond with the expected system response. Wizard of Oz is a method to use when wanting to test concepts before diving fully into the development of a solution.

Thinking Aloud

Thinking aloud is a method used to understand what users think about the system by letting them say what are they doing and thinking during given tasks. This gives the designers an idea of what the users think about the design, therefore, gathering important information. Furthermore, the designers learn why some users find certain parts of the design confusing and difficult to understand, which can improve

the next iteration of the product. On the other hand, it can show which aspects are good as they are in the current state.

3.4.2 Lo-fi Setup

The projector, camera and table were set up as described in Section 3.1.4. However, the layout that each player saw projected on the table, during the Lo-Fi, also has to be explained.

This layout can be seen in Figure 3.13. On the left, is the grid where the players would place their ships in the beginning of the game. To the right, is the grid where they would attack the opponent. Instructions about the game will be displayed below the grids, in the information box to the left and history facts in the box to the right of it.

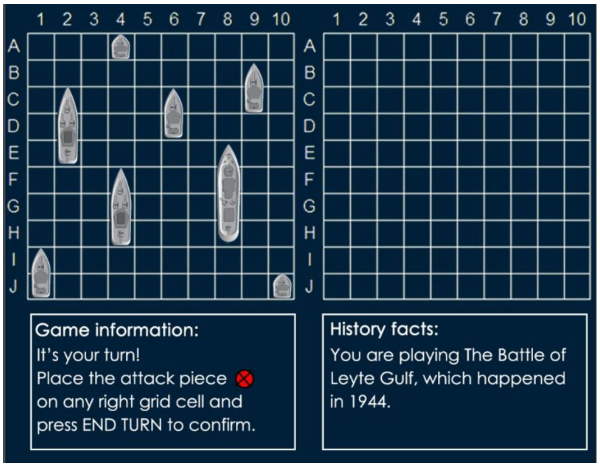


Figure 3.13: *Illustration of one player side, showing the grids and the text boxes placed below it.*

3.4.3 First Lo-Fi Test - Basic Gameplay

For this early prototype, the historical aspect was not taken into account. This decision was made due to the purpose of the Lo-Fi being to test the usability as mentioned in Section 3.4. This means that there were not focus on historical accuracy, as well as an accurate representation of the ships.

The beginning of the first test was done with internal usability testing. The internal testing was mostly done in order to test the side by side grids, as well as the number of ships for each player before any external participants would test the prototype. The Lo-Fi prototype contained the combination of the physical and digital aspects of the game, in order to see how well the participants adapted to

the combination. This would also more closely resemble the experience the participants would have, with a fully functional prototype.

In regards to the ships, it was decided that the 4x1 ship was efficient as the biggest ship on each player side, meaning that the 5x1 ship from the original Hasbro game was removed. The ships chosen for the Lo-Fi were as following:

- One 4x1 ship (Command ship)
- Two 3x1 ships (Battleships)
- Three 2x1 ships (Heavy cruisers)
- Two 1x1 ships (Destroyers)

This means that the players would have a total of 8 ships on their respective sides. These ships were, for the purpose of the Lo-Fi, made of paper, as seen in Figure 3.9.

Lo-Fi Data Gathering Procedure

There is more to a prototype than design. As mentioned in Section 3.4.1, the purpose of it is to test certain design choices so that improvements can be made, thus leading to reiterations. Accordingly, testing a prototype is a crucial aspect of the cycle.

For testing the usability, System Usability Scale (SUS) questionnaire [5] was used. Furthermore, were the participants interviewed with additional questions. This was presented to the participants after they tried the game.

The SUS questionnaire is a usability evaluation tool used to measure the usability of a certain product. The participants were told to answer 10 questions with answers based on a Likert scale. This is a scale commonly used in questionnaires, often used interchangeably with rating scales, ranging from zero to five. Here zero means strongly disagree and five means strongly agree. The score for the uneven questions is done by subtracting one from each answer and then summing those results. The even questions score is calculated by subtracting the answer from five, then summing those results together. The even and uneven scores are then added and multiplied by 2.5. The score received can then be compared to a scale, see Figure 3.14 which concludes how well the system's usability is.

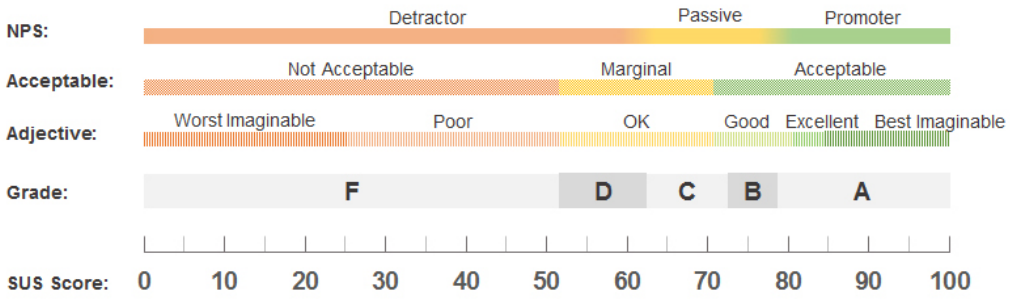


Figure 3.14: *The SUS scale [5].*

As mentioned, following the SUS, participants were interviewed. The purpose of the interview was to gain extra information or data that the SUS could not provide. To see the question contained within these tests, check Appendix A.

The Lo-Fi testing was conducted on five participants. The first participant served as a pilot study in order to evaluate the testing methodology.

All the interviews and the gameplay was recorded, in order to be transcribed and analysed. Before participating in the test all the participants signed a GDPR consent form, confirming that the conducted data can be used for this particular study.

First Lo-Fi Gameplay Procedure

For the creation of the prototype, design choices described in Section 3.3 were used. Game instructions were projected below the player's grid and changed throughout the game, according to what the player had to do. Below these instructions, outside the projection area, there were paper-based game pieces representing the ships. In the beginning of the game, the player would take the game pieces and place them where they desired on the left grid. For the purpose of Lo-Fi testing, a physical "End turn" button was placed outside the projection area. When the player was satisfied with their placement, they would press this button to indicate the end of their turn. At this point, the participant was instructed to remove the game pieces from the left grid, as there were now projections of the ships, in their place. To attack their opponent, they were instructed to place a paper circle in the desired position on the attack-grid and press the "End turn" button. Following that, they were informed whether they hit or missed by coloured circles on the respective grid cell, as illustrated in Figure 3.15. The game was played for a maximum of 10 minutes. They would, however, stop playing before that, if the participant sank all the opponent's ships.

For the purpose of the Lo-Fi, these functions were not yet implemented. How-

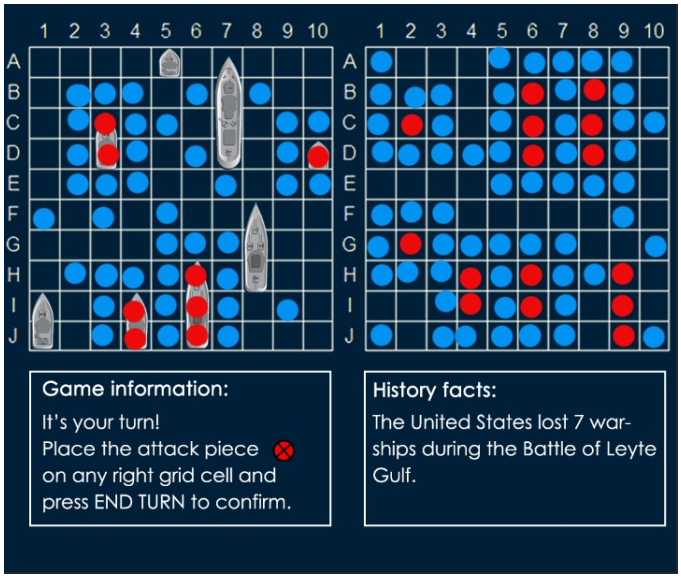


Figure 3.15: *Projected layout.*

ever, the goal was to create the illusion that the prototype did work as intended. The method called "Wizard of Oz" was used in order to achieve this. This method is described in Section 3.4.1. Furthermore, were the participants asked to "Think Aloud" throughout the test.

As seen on Figure 3.16, in order to give the impression of a working game two people would sit on the right side, opposite of the participant, taking the role of the opponent, but also controlling the game. Their job would be to respond to the participant's actions. This involved two tasks that they would have to perform. One would have to play a sound effect every time the participant pressed the "End turn" button. The other "Wizard's" job was to simulate the system itself by actively modifying the picture being projected using an image editing software. He would have to place the images of the ships, where the participant placed their pieces, as well as mark the misses and hits with coloured circles. Essentially doing the job of the program that has not yet been implemented.

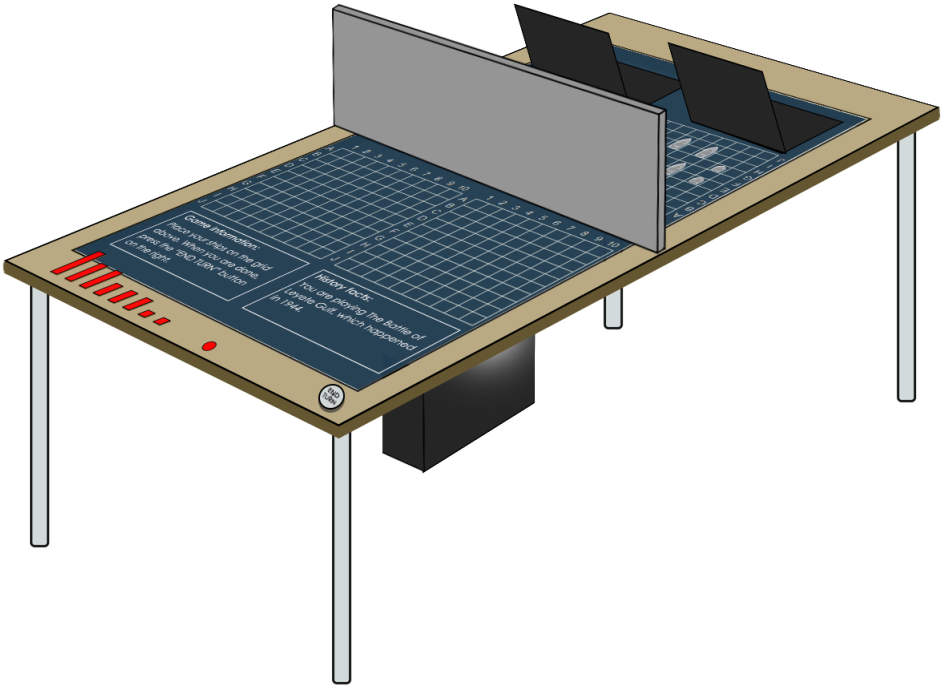


Figure 3.16: *Lo-Fi setup from the participant's side. The participant has the game pieces(red shapes) on the bottom of the projection. There is a physical button on the bottom right corner labelled "End turn". Two laptops are set up on the opposite side of the table for the "wizards" that simulate the game mechanics. The projector can be seen below the table.*

Results/Discussion

The SUS results are summarised in Figure 3.17. Line by line, the answers of all four participants can be seen to each question, the last rows containing the combined, total score for each participant. They have been calculated using the method described in Section 3.4.3.

Through using the SUS questionnaire it was found out that participant one, two and three scored 87,5. This score is equivalent to an "Excellent" rating. Furthermore, did participant four score a 92.5, which also is equivalent to an "Excellent" score on the SUS scale. This proves that the usability of the Lo-Fi overall can be considered a good rating [5].

All the answers to each question in the interviews can be found in the Full Appendix folder. The answers given to each question can be summarised as follows:

- **Did you notice anything changing throughout the game?**
  All the participants noticed that the historical facts changed during the game, however, they did not pay much attention to them, as they were mostly con-

SUS	P1	P2	P3	P4
1	4	3	4	4
2	1	1	2	1
3	5	5	5	4
4	1	1	1	1
5	4	3	4	5
6	2	1	2	1
7	5	5	5	5
8	2	1	2	1
9	5	4	5	5
10	2	1	1	2
Sum even	18	15	18	18
Sum uneven	17	20	17	19
Total	87,5	87,5	87,5	92,5

Figure 3.17: *System usability scale results.*

centrating on the game itself.

Participant 1: *"uuuh.. almost at the end, I noticed that the.. the information - the history information thing changed.. though I didn't really read it. Only once because I saw it changed."*

Participant 4: *"The facts, the history facts were changing."*

- **What do you think about putting the game pieces on the board and then taking them off again?**
  For all the participants except one, it did not pose a problem. The participants actually found it as a positive feature.

- **How intuitive did you find the game?**
  All the participants found the game very intuitive. They claimed that the Game information in a box helped a lot in the beginning of the game and afterwards they didn't need any assistance.

  Participant 1: *"Very intuitive"*

  Participant 2: *"It was pretty intuitive..." "...after one turn I really understood it, yeah"*

- **What are your thoughts about layout?**

The feedback on the layout was mostly positive. One participant suggested to have his grid slightly larger than the grid to shoot at. Another suggestion was to make the game pieces 3D instead of pieces of paper.

Participant 2: *"my first thought was to have my grid a little bit.. Bigger than the opponents but... I don't know if that makes sense"*

Participant 4: *"the only thing I would do maybe is like make them (the game pieces) like 3D, not like just pieces of paper. That would be like really nice."*

- **What do you think about the colour scheme?**
  The colour scheme was well-received, the participants claimed that the colours blue and red for miss and hit respectively made sense to them and that it felt natural.

- **Any thoughts on visual and audio feedback?**
  The audio feedback was the main issue that the participants had. Even though the presence of the sound that ended turn was appreciated, it was said to be repetitive and annoying. It was also mistaken for a sound that indicates miss or hit. It was suggested to change the sound and to have different sounds for different actions.

- **Any additional comments?**
  Overall the Lo-Fi was praised. One participant commented, that it might be tiring to stand for more than one game at a time.

  Participant 3: *"No, not other than I liked it, you have done a really good game."*

In conclusion, the Lo-Fi testing can be considered successful and brought valuable feedback. The setup proved to be working, as the players were not able to see the other side of the table at all, while still being able to see the other player. The projection of grids and text on the table was clear. The players found the gameplay intuitive and were fully invested in the game and even entertained. Teaching history was not an aspect that was tested during this testing. However, it included historical facts being presented. The overall score on the SUS scale ranged from 89,5 to 92,5 which translates to an "Excellent" score.

3.4.4 Second Lo-Fi Test - Power-ups

This test of the prototype was only tested internally without participants. The goal was to find a balance, where the implemented power-ups would not make the game less strategic, as well as replace the purpose of a regular attack.

Second Lo-Fi Gameplay Procedure

This procedure was more similar to the original version of the game Battleship, since the board itself was physical, made out of paper and the placed ships would not be removed after being placed by the players as in the first conducted Lo-Fi. The attacks would be processed through verbal communication (e.g. saying "B4") instead of using a button. The hits and misses would be marked with small paper pieces (a blue piece for a miss and a red piece for a hit). Furthermore, wooden pieces were used as ship pieces, as described in Section 3.3.3. The new power-up pieces were all made out of paper in order to be easily adjustable if found necessary. Figure 3.18 shows an illustration of these changes.

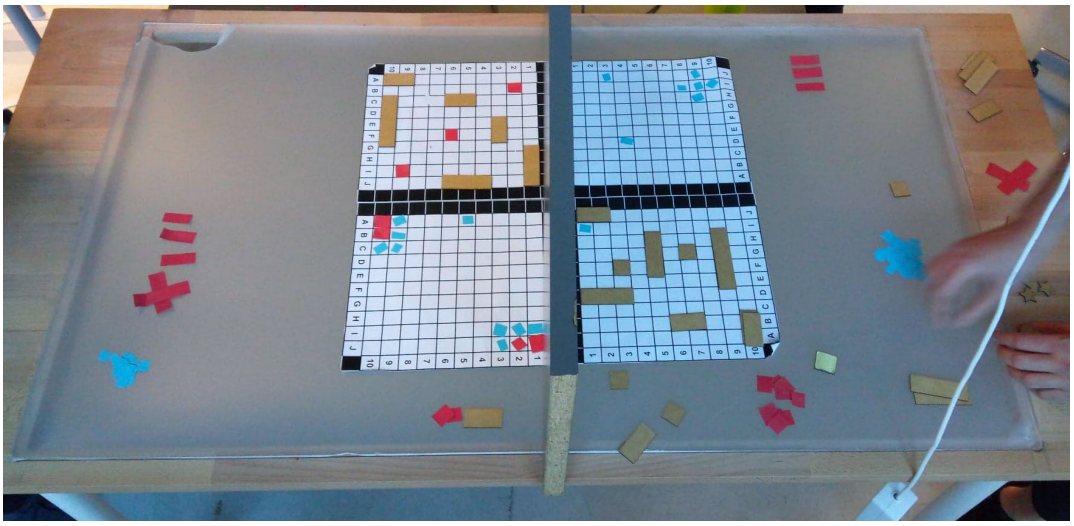


Figure 3.18: *An illustration of the set-up during the second Lo-Fi test.*

Two people were playing at the same time, instead of one player, playing against "Wizards" as in the first Lo-Fi test. The "Wizard of Oz" method was, however, still used when the power-ups were used. The usage of power-ups was tested over two games where the players would have three 1x1 pieces, from which they could choose how many of those should be decoys or mines. In another game, the players had the ability to choose either two decoys or four mines as described in Section 3.3.2. In either version, the Wizard would keep track of whether the placements of the mines had hit any of the opponent's ships, as the game started. Additionally, the Wizard controlled the random hits from the Aircraft Carrier power-up, by dropping a 1x1 game piece onto a random position on the attacking player's attack-grid. Furthermore, the Wizard kept track of the number of turns the Aircraft Carrier had been running for, as well as the cool-down time for it.

This procedure would continue until one player would win the game. Follow-

ing this, the power-up's functions were discussed and changed if needed. In case something was changed, the game would be played again, to test out the new concept.

Results/Discussion

Through testing the power-ups, it was concluded to use four of those power-ups mentioned in Section 3.3.2. These power-ups were the **Torpedoes**, **Aircraft Carrier**, **Barrage** and **Mines/Decoys**. The Radar was considered to cover too large an area, in combination with the other power-ups and because of this, it was decided to scrap the idea. The Last Stand was also considered to cover too large of an area and therefore also removed. It was also considered to lower the number of random tiles this power-up would hit, however, the combination of the four that was ultimately chosen seemed sufficient. This may, however, still change in later iterations of the game, or be different depending on which battle the players chooses, in a more advanced version of the prototype.

Chapter 4

Final Problem Statement

Based on the knowledge obtained throughout the background research in Chapter 2, it was found that a form of aid could be used, in regards to teaching pupils in the age group of 12–19 history. The proposed solution is a combination of a board game and a video-game, as it can benefit from the advantages of both of them. This would hopefully help teach the pupils of said age, and motivate them to learn. That being said, a final problem formulation is created:

> ”Learning history is not motivating for most pupils within the age group 12–19 years old, how can we make the process more engaging through a digitised version of an existing board game, projected onto an interactive table?”

With this problem formulation in mind, an actual prototype is to be established. To ensure the best possible outcome of the implementation, some system requirements have to be created, in order to set boundaries.

4.1 System Requirements

The solution must fulfill the following specifications:

- A computer running Linux or Mac OS with the created program.
- A table with a surface area of 155 x 75 cm, and a height of 93 cm. The table has to incorporate a semi-transparent surface, with an area of at least 125 x 69 cm, which accommodates the projection. The projection must be clearly visible from above the table, meaning that the text displayed should be readable, and as little distortion or blur as possible should be present.

- A short-throw projector that can project an image of approximately 125 x 69 cm, at a distance of 62 cm from the table surface.
- A top-mounted camera that can capture an area of at least 155 x 75 cm, when placed 170 cm above the table surface, looking down at the surface.
- A thin (approximately 0.5 cm), 35 x 71 cm wall placed in the middle of the table, separating it in two equally sized halves.
- Game pieces equal to or slightly smaller than a grid-size of 4x1 (command ship), 3x1 (destroyer), 2x1 (heavy cruiser), 1x1 (mines/decoys) and a piece fitting for the barrage power-up (two 3x1's connected at the middle forming a cross). With previously mentioned dimensions of the individual system components, a piece that fits one grid cell should be 23 x 23 mm.
- The system must be able to detect the physical game pieces of said dimensions from a distance of 170 cm using the camera.
- The system must be able to detect four 10 x 10 grids, each being approximately 25.7 x 25.7 cm, from a distance of 170 cm.
- The system must be able to distinguish which particular grid needs to be used for the current state of the game (e.g. player one attack, player two attack or game setup phase).
- The system must be able to distinguish and correctly allocate the game piece's location within the four 10 x 10 grids.
- The player should receive visual- and/or audio feedback, informing whether their attack was successful (hit) or not (miss).
- The system must be able to end/confirm the player's turn by detecting a hand gesture at a designated mark.
- The system must incorporate history teaching, by using the "Interaction with historic themes, concepts, choices and resources" and "Play within a historical or history-related setting" historic game types.

Chapter 5

Design and Implementation

This chapter revolves around the considerations made in order to implement the designed system from chapter 3.

5.1 Theory of Methods

In this section the image processing algorithms and methods used for the implementation will be covered.

5.1.1 Colour Thresholding

Thresholding is one of the most fundamental point processing operations, which is used for replacing each pixel in the image with either a black or white pixel. This is depending on pixel intensity constant and its comparison to the input of the original pixel value, hence turning into a binary image. Thresholding can be described by the following segmentation algorithm:

```
if f(x, y) <= T then g(x, y) = 0
if f(x, y) > T then g(x,y) = 255
```

Where function f is the input image, T is the threshold value and g is the output image. The algorithms mentioned above simply says: *"if the input image's pixel is less or equal to the threshold, then the output image's pixel is black, else it is white"*. This is most commonly used for grey scale images, however, it can also be used to threshold colours as shown in Figure 5.1.

This can be used to segment out objects of interest from a image, which makes it good for object detection. However, instead of thresholding the RGB (red, green and blue colour space values), it is more common to threshold the hue from the HSV colour space instead. HSV stands for hue, saturation and value. The algorithm looks accordingly:

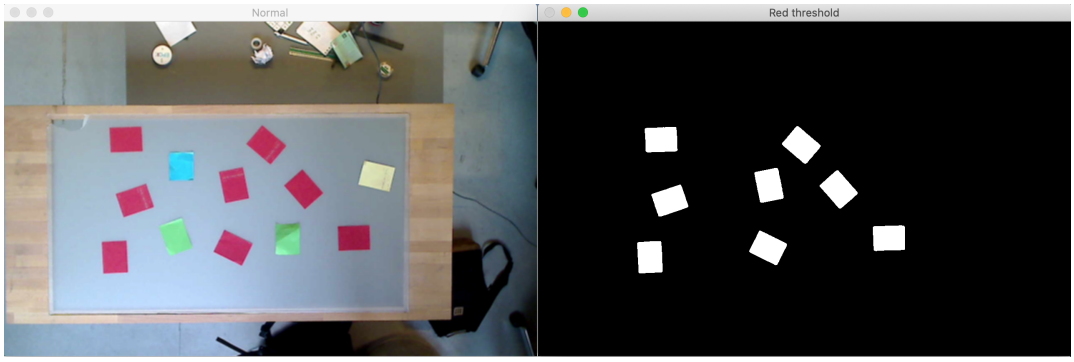


Figure 5.1: *Red threshold applied to mask.*

> Hue = if f(x, y) > lower bounds and f(x, y) < upper bounds then g(x, y) = 255 else g(x, y) = 0

An example would be wanting to threshold the colour red:

- Find corresponding HSV values from the inputted RGB value.
- Set upper and lower threshold.
- Mask the image based on the thresholds.
- Display new image.

5.1.2 Background Subtraction

Background subtraction is a technique used for finding changes in a frame e.g., moving objects by segmenting the foreground from the background, hence only getting a frame with the changes on. This makes it good to use for solutions which rely on isolating movement from a scene. The background can be perceived as a mask that gets applied on top of the foreground only making the changes visible, however, if the images has different lighting, interiors, exteriors or noise it can also be perceived as a change hence resulting in more changes then wanted.

Background subtraction can be achieved in two ways, either by using a local threshold or a global threshold. **Global threshold** is when the same threshold is used for every pixel, hence making it "global". This is very inconsistent since illumination changes can occur in the foreground which will result in different histograms that will yield a large variation when comparing pixels. A solution to this is using a **local threshold** which will create a unique threshold for each pixel dynamically. This is better to use if the conditions of the images will vary, hence

taking into consideration the rapid changes that can occur. To create the binary image the following equation [18] can be described:

$$BinaryImage = if\, Abs(g(x,y)) < \beta * \alpha(x,y) \quad \text{then} = 0, \quad \text{else} = 255$$

Here **alpha**(x, y) is the standard deviation and **beta** is the scaling factor. Beta will remain static throughout the whole process but alpha will change for every pixel. If the abs is lower than standard deviation multiplied with beta the pixel will be 0, otherwise if its false it will become 255. This will create a binary image which only consists of pixels that are either 0 or 255.

5.1.3 Morphology

This part of the design and implementation section will cover a branch of Morphology, more specifically Erosion. Morphology is a neighbourhood processing method, used on gray-scale and binary images. From an input a kernel is applied for each pixel and denoted a structuring element, so that it contains 0's and 1's. From here one of two methods in a three level operation as can be seen in Figure 5.2 is applied known as Hit or Fit. Fit is the first level of what will turn into the second level known as Erosion [18].

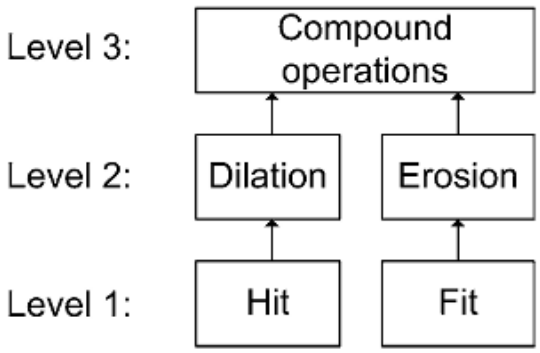


Figure 5.2: *The three level operation of Morphology [18].*

When applying fit it is investigated if the 1's positioned in the structuring element are the same as the 1's in the image as can be seen in Figure 5.3. If this is the case for every structuring element then it is classified as fitting the image and the pixel is set to one in the output image. However if this is not the case, then the pixel is to zero [18].

Below is the formula of Erosion, here the output image pixel is equal to the input pixel with the kernel SE applied [18].

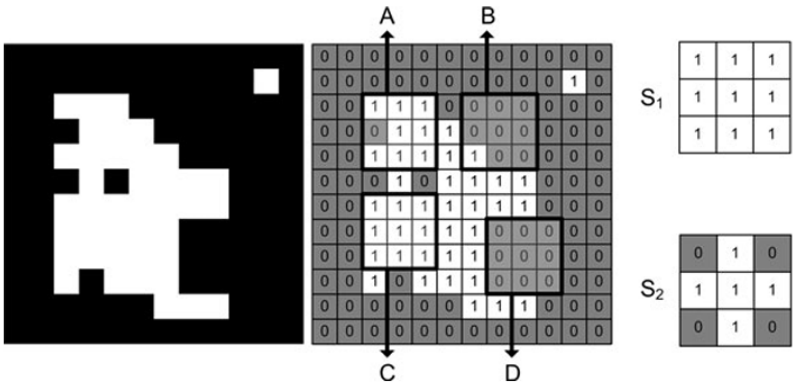


Figure 5.3: *An example of how a binary image can be compared to a structured element, where S1 and S2 are representing the structured elements [18].*

$$g(x,y) = f(x,y)\theta SE$$

5.1.4 BLOB Analysis

BLOB is a backronym and stands for binary large object, which are regions of an image that looks somewhat coherent. This essentially means that it can be used to find objects of interest in an image and then afterwards do some processing on them. This could for instance be removing, labelling or compositing them.

For finding the BLOBs, an connected component analysis has to be carried out on an binary image. This means that the image of interest has to be processed through a threshold to turn it binary before the BLOB analysis can be achieved. BLOB analysis is carried out by using neighbourhood processing on each pixel in the image and for this two kernels can be used.

Figure 5.4: *The 2 kernels; 4-connectivity or 8-connectivity [18].*

As seen on Figure 5.4 the two kernels describe the connectivity of an object. If the 4x4 kernel is used, objects connected on a diagonal level will not be perceived

as one object, where in contrary if the 8x8 was used they would. After the kernel has been decided the actual BLOB analysis can start.

Given an input binary image the process will start in the top left corner and iterate until the last pixel is met in the bottom right corner. If a white pixel is hit the program will apply the kernel with the pixel as the centre. Then it will look at all the positions in the kernel (neighbourhood pixels) to see whether it is connected to any white pixels. If it is a white pixel it will mark them and then afterwards burn them (set them to zero). This will continue until it has been done with every white pixel, which will result in BLOBs labelled depending on the amount e.g., if there were three BLOBs it would be labelled as one, two and three.

The algorithm can be described as the following:

$$if f(x,y) > 0 \quad \text{then} f(x,y) = 0 \quad \text{afterwards check neighbour pixels.}$$

After the initial pixel is not white, it will burn it by setting it to 0 and then afterwards checking all the neighbour pixels for the same condition. Every time one of the neighbour pixels meet the condition it will add them to a queue. Then the program will remove the last element in the queue until there are no more pixels to check, thereby ending the blob analysis.

5.1.5 Gaussian Blur

Gaussian blur is the result of blurring an image by applying a Gaussian function to reduce noise or details in an image. It is often used in connection to computer vision to improve the structures of an image at different scales. To calculate the transformation that is to be applied to each pixel of the image, the following formula is used:

$$G(x,y) = \frac{1}{2\pi\sigma^2}\epsilon^{-\frac{x^2+y^2}{2\sigma^2}}$$

The G in the formula is the resulting image. The x and y are the distances from the beginning of the horizontal and vertical axes respectively, in other words, the specific pixel for which the new value is calculated. The σ is the standard deviation of the Gaussian distribution. The distribution of individual coefficients in different sized kernels, when the formula is applied, can be seen on Figure 5.5.

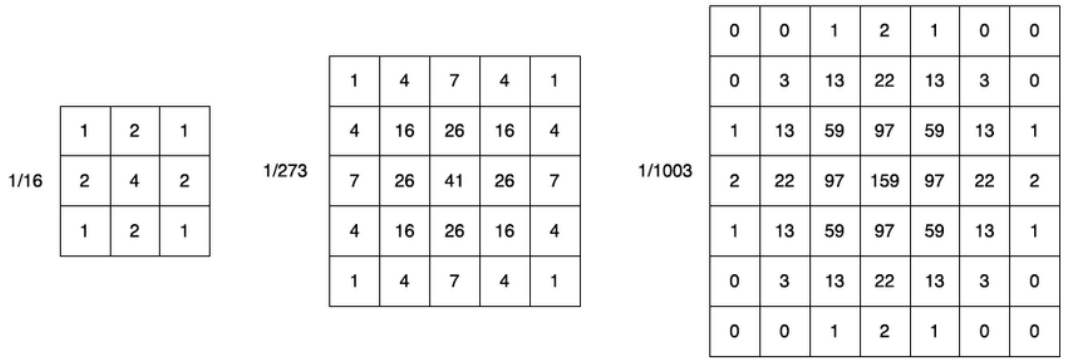


Figure 5.5: *Three kernels showing different sizes of Gaussian blur.*

5.2 Final Design and Implementation

This section will explain how the final system was implemented, what modules it contains, and how they interact with each other.

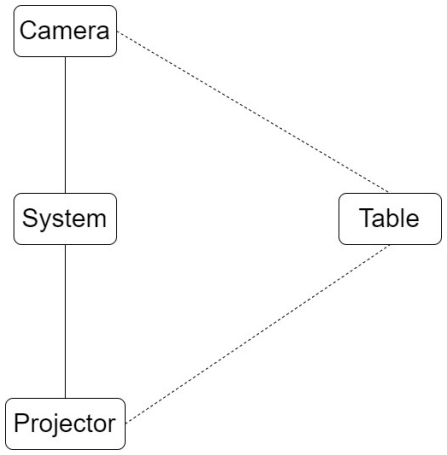


Figure 5.6: *A diagram explaining the setup elements required for the proper functioning of the system.*

Firstly, the way each element in the set-up interacts with each other has to be explained. As can be seen in Figure 5.6, the system itself is only directly linked to the projector and camera. For the system to function however, it needs input from the camera, either in the form of images or video. This input is captured from the table.

The system is also connected to the projector, which projects, onto the table, the screen of the laptop on which the system is running on. This output image is modified and changed based on the camera inputs taken earlier. And so, all the elements seen on Figure 5.6 are dependant on each other and work together to create a fully operational interactive table.

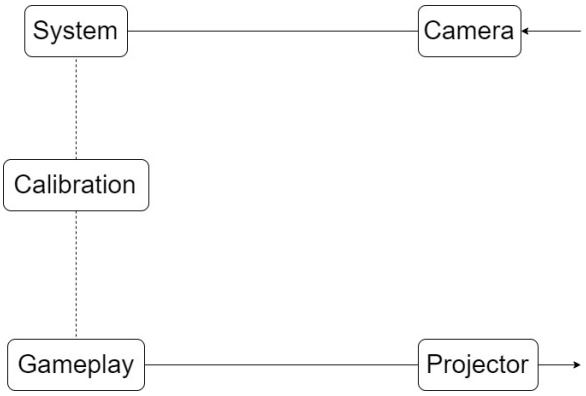


Figure 5.7: *A small expansion of the system part of the set-up.*

Furthermore, the workings of the system can be broken into two steps, the first one being a calibration phase and the second one being a gameplay phase, as can be seen in Figure 5.7. These two phases will be described in greater detail in their own sections below.

This module has the purpose of finding at which coordinates the grids are located in the input image. This information would then be sent to the Gameplay module, where it would be used to crop the grids and detect the appropriate position of certain shapes relative to the grid. So, in order for its purpose to be fulfilled, it needs to accurately and consistently find where the grids should be located.

Implementing the Calibration Module

Keeping its purpose in mind, the first method of implementation tried proved to be inconsistent. This relied on display four cubes at each corner of a grid, as seen on Figure 5.8. These cubes would then be found using colour thresholding, as described in Section 5.1.1, at which point either their top right or bottom left would be used in later parts of the code. The advantage of this was that one could always run this module whenever in order to find the coordinates of the grids at that particular time. This meant that even if the table was moved the program could be adjusted to that movement.

However, as previously mentioned, this approach proved to be very unreliable. This was caused on one hand by the fact the cubes were quiet small and that made it difficult to eliminate noise after the thresholding without risking to lose the cubes as well. On the other hand, the brightness of the projection, even when turned to its lowest setting, made it so that some of the cubes would blend in too much with other elements of the game screen. This is why, when trying to apply

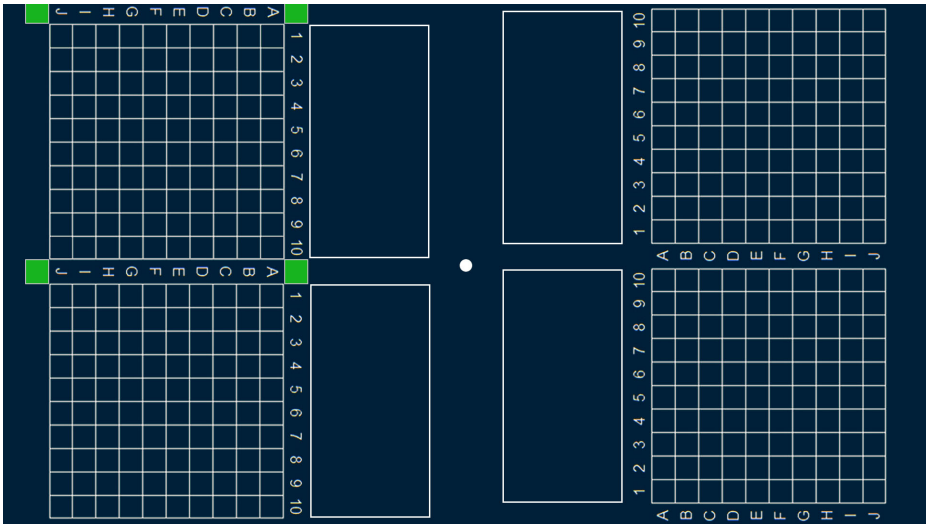
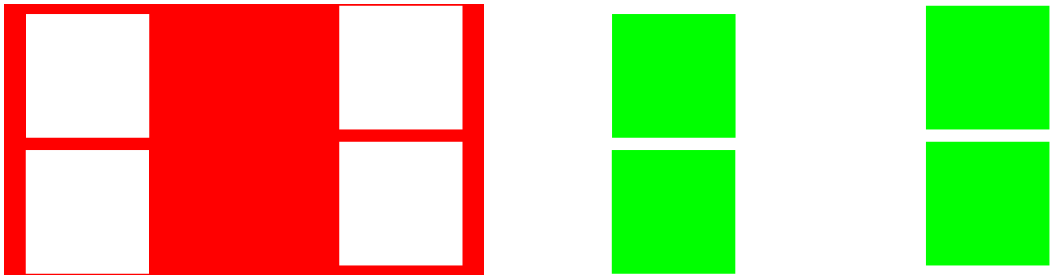


Figure 5.8: *Initial BLOB detection picture.*

colour thresholding, described in Section 5.1.1, the results would vary a lot, and a threshold that was accurate for all cubes could not be found.

Following this, a new approach was tried. As a start, the information boxes were moved above the grids, due to the brightness of the projector mainly obscuring the centre of the table. As nothing in the information boxes needed to be detected by the system, it was decided that moving the grids further away from the centre was a better approach. Additionally, instead of displaying the actual game screen, a background containing a single colour would be displayed. On this background there would be four big cubes that all had the same colour which was distinct from the background. These cubes would have the same size as the grids and be located in the same spot in the image.



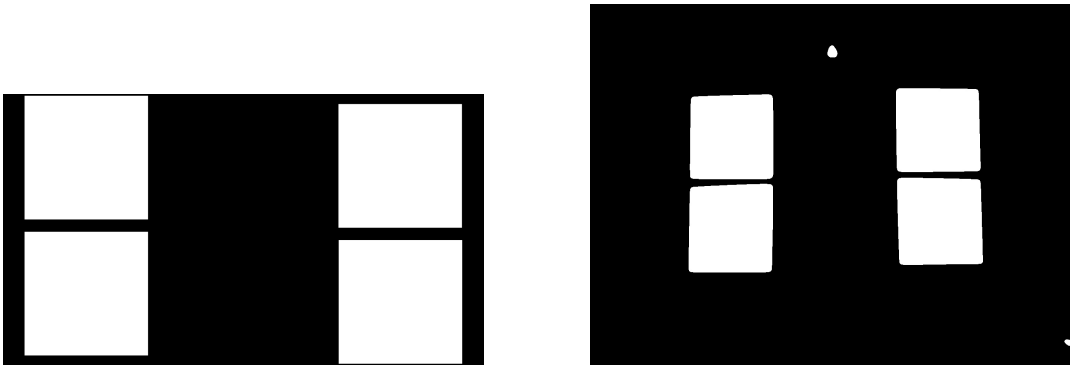
(a) *Red background with white squares.* **(b)** *White background with green squares.*

Figure 5.9: *Calibration test pictures.*

Initially, different combinations of background and square colours were tried.

The examples can be seen in Figure 5.9. However, it was found that having a light colour as the background was inefficient since the same problems as before would arise, the threshold was not consistent enough for the entire game screen and the squares would blend in with the background sometimes when trying to apply it. As such, it was decided to turn the background black, this would eliminate the problem of there being to much light on the game screen, as well as create a higher contrast between the squares and the background.

Furthermore, a lot of the noise created because of the brightness of the projection and multiple colours was eliminated. In Figure 5.10a the image of the four squares on a black background can be seen. After this image was displayed on



(a) *Original image.* **(b)** *After the threshold has been applied.*

Figure 5.10: *Before and after calibration.*

the table using the projector, the camera would capture it and send the picture to the system. The system then would use colour thresholding to find the white or very light gray, after applying a Gaussian blur, described in Section 5.5, a visual representation of the logic of this module can be seen in Figure 5.11. A result of how the input image would look after these processes can be seen in Figure 5.10b.

5.2.1 Calibration module

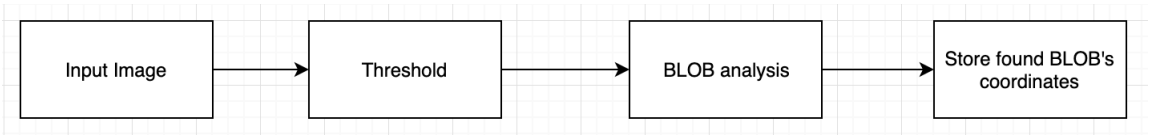


Figure 5.11: *Calibration module logic.*

In order to eliminate any extra noise, BLOB extraction was used on the processed image in order to find all the BLOBs in the image. Among those, of course, would be the four big squares. The BLOBs that were too small or too large would

then be filtered out, and the only ones left would be the BLOBs representing their grids. The minimum and maximum coordinates of these BLOBs would then be extracted and they would be then used later in the code. This approach proved to be much more reliable than the previous one and therefore it was kept. It still has its disadvantages however. It not only adds an extra step before actually starting the game, but it also falls apart in case the table or the camera is moved after the first coordinates were stored.

5.2.2 Gameplay

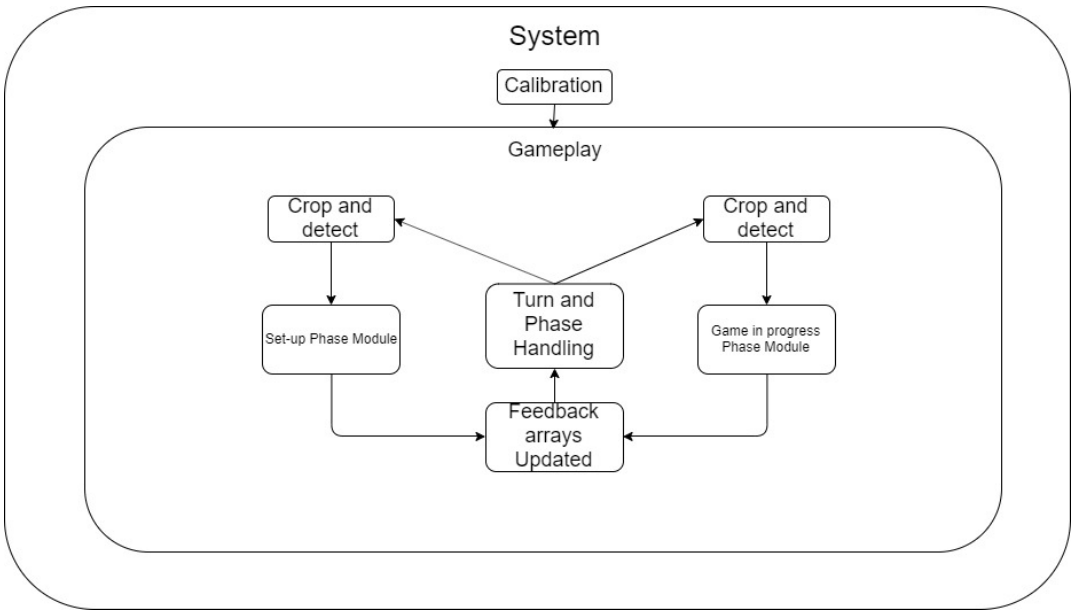


Figure 5.12: *The expansion of the gameplay phase into multiple modules.*

The Gameplay phase can be split into multiple, smaller modules which, when combined, ensure the proper functioning of the system. These can be seen in Figure 5.12. Each of these modules will be explained separately below.

Crop module

This module's purpose is to crop the image into four grid images based on the coordinates received from the calibration module. It will crop the grid image both with and without ships on and store them in two arrays that will be used for background subtraction later. Once the images has been stored the program can then proceed to the other modules.

Implementation of the Crop Module

In order to crop the grid image this module takes the coordinates stored from calibration and slice the image four times (one time for each grid). The coordinates which are used can be seen in Figure 5.13. It does so by first taking an image of the current frame of the camera and save it into the program. Images in python is represented as either 2D or 3D arrays consisting of pixels, they can then be referred to as arrays. Then afterwards the image array is sliced as following; image[y:y1, x:x1] where x, x1, y and y1 is replaced with the coordinates stored from calibration. It does so over two iterations; the first iteration the players are prompted to press "q" without any shapes in the picture which will make the program take an image. Afterwards that are prompted to press "e" to take an image with shapes on. After the two iterations the program then saves the eight images into two arrays where the first array is the pictures without shapes and the second one with shapes.

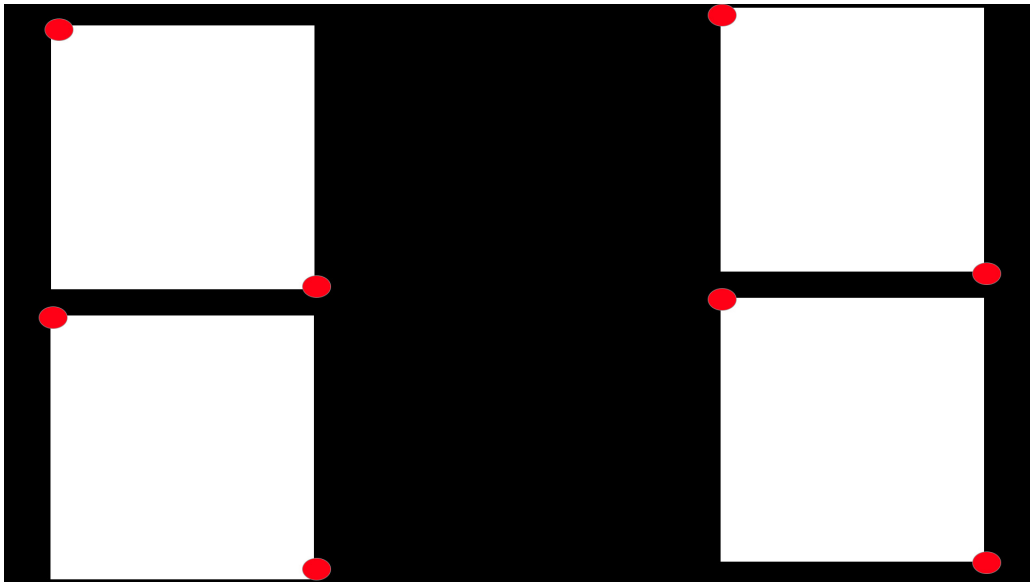


Figure 5.13: *A representation of what coordinates are used in order to crop the grid, the coordinates are represented by red circles.*

Detection Module

The purpose is to find the ship placements based on the stored images and return the coordinates of the ships to the next module. It does so by applying mathematical logic and operators to determine where the ships are placed within each respective grid.

Implementation of the Detection Module

To detect the game pieces and their location on the interactive table, a global threshold was first attempted as a method to detect the game pieces. However because of different lighting conditions on the table, it was an inconsistent method to use in practice since the camera often would detect more BLOBs than intended. In an attempt to solve this issue the gamma of the colours were increased with the hope that it would be easier for the camera to detect the game pieces and the game pieces only. However the lighting conditions were still affecting the threshold and the results were therefore similar to the those received without modifying the gamma. And so, the idea to use colour thresholding was dropped. Instead, background subtraction was used. The program would take the image of the background, using the crop module described previously, as well as the current state of the grid, with the shapes on, again described previously in the crop module. It would then calculate the absolute difference between the two images, after they were both turned to grayscale. Initially, histogram equalisation was also used in order to create a higher contrast between the pieces and the rest of the grid, however this was found to be to inconsistent in its results when even small light changes would be caused. Finally it was decided to just use the images taken, as they were. The extra noise created when subtracting the background would be eliminated first by applying a higher threshold when turning the image binary, which would get rid of small differences between the two images. Secondly, erosion was applied which eliminated the small noise that was left.

To detect the position of the game pieces within the grid, the cropped image of the grid, taken in the crop module, is used. This image is then divided in tens, both vertically and horizontally. This would result in the image basically being split into a 10x10 grid, similar to how it looks in reality, but now, the program would have the actual coordinates of each line forming the grids, a representation of this can be seen in Figure 5.14. This way of detecting the grid positions makes the program adaptive since the it does not require the image to be clear or cropped exactly right, allowing for small deviations that might happen in the crop module.

After detecting the piece on the board using the background subtraction mentioned earlier, and eliminating the noise, the program then takes the minimum and maximum coordinates of the detected shape and stores them for comparison against the grid. It then loops through each vertical line, finding both the maximum line (coordinate wise) that is still smaller than the minimum coordinate of the shape, and the minimum vertical line (coordinate wise) that is still bigger than the maximum coordinate of the shape. A short explanation of this algorithm can be seen in the box below. The program then does the same horizontally. As such, if a person was to place a ship of size two, occupying coordinates [2,3] and [2,4], the program would detect that it is vertically between line 2 and line 3 and horizontally between line 3 and line 5.

$$x = 0, if\, x * ((imagewidth)/10) < minimumxoftheshape, then x++ \quad (5.1)$$
$$x = 10, if\, x * ((imagewidth)/10) > maximumxoftheshape, then x-- \quad (5.2)$$

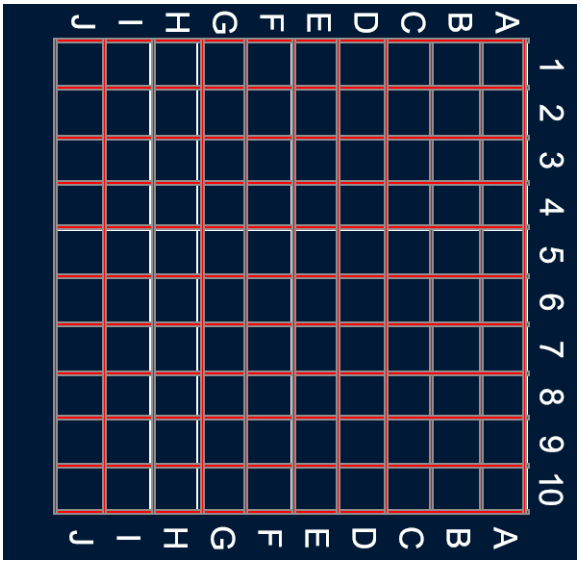


Figure 5.14: *This is a visual representation of how a grid would get divided based on the detected coordinates (represented by the red lines).*

After detecting which lines the shape is contained by, the program then has to convert these into actual coordinates on the grid. As such, first, it is checked whether the shape is place horizontally or vertically. If the difference between vertical lines is one, then the shape is horizontal and vice versa. Using the example from before, if a player has their ship at coordinates [2,3] and [2,4], then the vertical lines containing it are 2 and three. This then means the shape is arranged horizontally. After checking its orientation, the program then has to verify two cases:

1. If the shape is horizontal: the program takes the minimum vertical line and uses it as the x coordinate. It then starts with the minimum horizontal line and loops through the lines until it reaches the maximum horizontal line and uses those as y coordinates.

2. If the shape is vertical: the program takes the minimum horizontal line and uses it as the y coordinate for the shape. It then starts with the minimum vertical line and loops through the lines until it reaches the maximum vertical line and uses those as x coordinates.

Using the previous example, we assume a shape was between line two and line three vertically and horizontally between line three and line five. The difference vertically is one, which means the shape is horizontal. The program then takes the minimum vertical line, according to the algorithm, which in this case is two, and sets it as a x coordinate. It then starts at the minimum horizontal line, three, yielding the coordinate [2,3]. It then keeps looping until it reaches the maximum horizontal line, which is five, meaning the only other coordinate yielded is [2,4].

Setup Module

This module handles the beginning of the game. Here both players will have to place their ships, mines and decoys and then both will have to confirm their choice for the game to start. The purpose of this module is then to properly store the given information and handle the results of this preparation phase, so that the game will begin as intended.

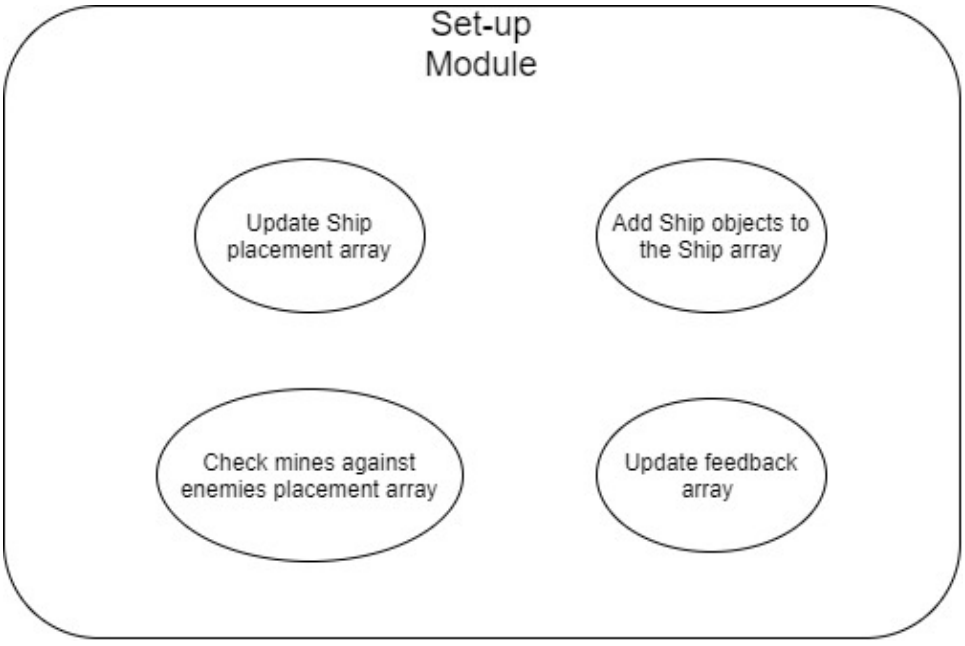


Figure 5.15: *A diagram explaining what the main functions of the set up phase are.*

Implementation of the Setup Module

As such, this module has to accomplish four things, those can be seen in Figure 5.15. Firstly, it updates a two dimensional array called placement. This is a copy of the grid but represented as an array of integers, where positions occupied by a ship on the real grid should be marked with a 1 and positions that are unoccupied

should be marked with a 0. This array is created using the coordinates received from the detection module.

The set up module also has to create Ship objects, which could then be used in later parts of the code. For this, it uses the Ship class. The attributes and functions

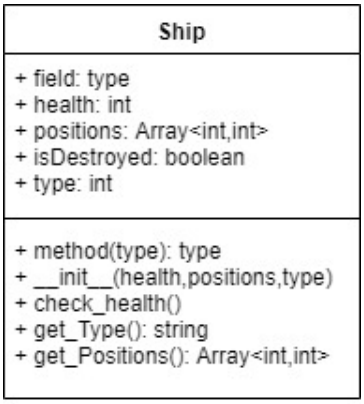


Figure 5.16: *A UML class diagram of the Ship class.*

of this class can be seen in the class diagram in Figure 5.16. It is important to know that the positions array described in the figure is an array that holds pairs of coordinates representing which position this ship occupies in the placement array. The health represent how much of the ship has been hit (in the beginning it is equal to it's size), the type represents the size of the ship, in regards to the grid. Finally there is a boolean that is either true if the ship is still operational or false if the ship has been completely destroyed. The created Ship objects are stored into an separate array for each player.

Finally, the mines, which are placed on the attack grid, are then checked for a hit or miss. The coordinates of these mines, grid wise, are received from the detection module. Furthermore, they are then subtracted from nine, and a new pair of coordinates is created.

The reason this happens is that a player would start the numbering of their grids from a different point, depending on which side of the table they would play. Normally, they would start numbering they grid from the top left corner. However, since the two grids are on opposites sides of the table, these centres will not be on the same side for both of the players. Where player one's centre may be on the top left, from his perspective players two's centre would actually be on his bottom right. This means that a mine at positions [1,2] on one players grid, actually needs to hit positions [8,7], if all players played the game from the same perspective. This is exactly what the camera does however, since it captures the image from a birds

eye view so all the shape positions are relative to the same corner, the top left. And so, the adjustment of deducing the new coordinates from nine is made. This ensures that the mines will be checked against the right positions

After the subtraction is completed, these new sets of coordinates are checked against the enemies placement array. If, at those position in the placement array, the number held is 1, then the mine hit, and the feedback array (a two dimensional array, where 2's means hits and 3's mean misses) gets updated.

Game in Progress Module

This module, as can be seen in Figure 5.17, is mainly supposed to handle the attacks within a game, by updating the feedback array based on whether a player has hit or not, as well as modify a ships health, in case it has been damage.

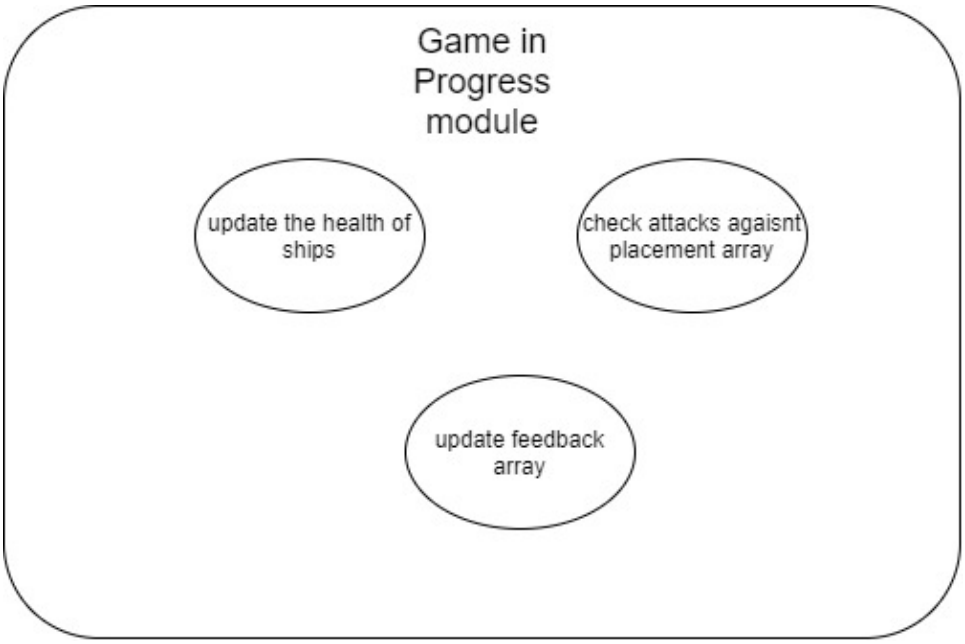


Figure 5.17: *A diagram detailing the main functions of the game in progress module.*

Implementation of the Game in Progress Module

This module operates in similar fashion to the checking of mines described in the Setup module. It too, converts the coordinates of the attacks received from the detection module, and check them against the opponents placement-grid. The difference is that this module also considers power ups and check them appropriately. The feedback array is also updated as a consequence of this module. This module

also updates a ship's health in case it has been damaged.

Visual Feedback

This module will project the visual feedback onto the image displayed for the players, this includes projection of ships, mines and attacks. It does so by taking the information in the previous modules and updates the image regularly.

Implementation of the Visual Feedback Module

After the players place their ships, decoys, and mines and the program creates the appropriate placement and feedback arrays, the program updates the image displayed onto the table based on them. It does so by using the library "Pygame[1]" which allows images to be modified in the sense that you can put layers on top of the image. By looking at the ship placement array attained from the detection module this module will take the coordinates and update the surface image used in pygame. It does so by getting the initial coordinate and furthermore applying an offset that will make it fit in the tile more properly. It does the same for the attacks, the only difference being that it looks at the feed array instead.

Ending Turns Module

In order to make the game turn based it is necessary to set up some conditions for the game to check, so the program can determine, which player is currently taking their turn and when should it end so that the other player can take their turn.

Implementation of Ending Turns Module

Because of the functionality of the game, player one and player two both have to end their turn to start the game, which will also confirm the positioning of ships plus mines or decoys. When this is done, the first player can take his/her turn, which will be the person classified as player one. The player have to place an attack piece or a power-up on the attack-grid and then press the 'Q' key on the computer keyboard to end their turn. If player one manages to hit an enemy ship or decoy (if used), then player one will be able to take another turn. However, if player one end their turn and misses or uses the barrage power up, it will then be the turn of player two. Player two take turns the same way as player one, but in order to end their turn they would have to press the 'E' key on the keyboard. When a player have hit all of his/her opponents ships it will no longer be possible for either of the players to take their turns, unless the game gets restarted.

[1] An open source python library used for making multimedia applications

Chapter 6

Testing and Evaluation

The testing and evaluation of the prototype will be split up in multiple iterations. The first, which will be described in this chapter, focuses on how well the system meets the system requirements. These tests were without focus on the historical content and the accuracy of it. However, future iterations of the prototype will focus on this, as well as e.g. usability and its performance when used by its target group.

6.1 Purpose

As mentioned, the purpose of this iteration was to see how well the system meets its system requirements. These are the system requirements listed in Section 4.1. This e.g. means that the tests focused on how consistently the requirements were met and if some of the implemented solutions were faulty. If that is the case, a new revaluation is going to be required, where these faults are going to improved upon. Four things were tested: The accuracy of the grid detection, the accuracy of the game piece detection, the accuracy of the localisation the game pieces within the grids (tested both on the ship pieces and the mine/decoy pieces) and the accuracy of the visual feedback displayed throughout the game.

6.2 Procedure

The tests were conducted in smaller "modules". This means that the specific parts e.g. detecting the game pieces (objects) or their location within the grid were tested separately. Each module was tested multiple times to see if the results were consistent. Looking at the system requirements, it was possible to determine what to test and how to set up the tests, which were all conducted internally. Throughout the process, there was one person responsible for controlling the system, a second for interacting with the table itself and a third for documenting the process.

6.2.1 Grid Detection

To test the grid detection the game was run like a normal game. However, two additional tests were conducted. After calibration, two tests were performed: one where the table was slightly moved, and one where the projector was moved.

6.2.2 Game Piece Detection

For testing the game piece detection, a single 1x1 game piece was used in all four grids. The game piece was physically moved to every single grid cell, as seen on Figure 6.1. The system would save an image of each position and process it, as explained in Section 5.1.3. This process was done twice for each grid.

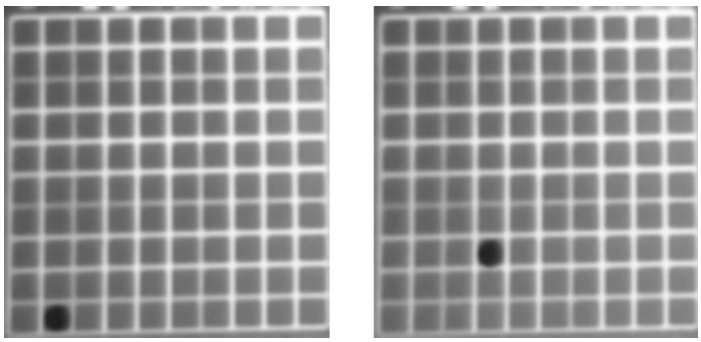


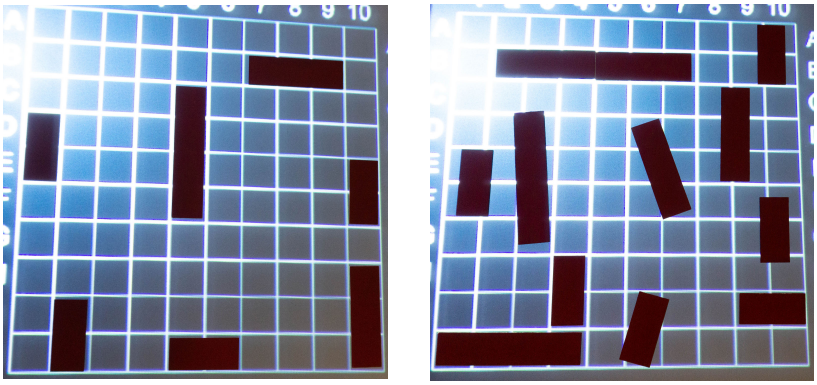
Figure 6.1: *Images from the game piece detection test.*

6.2.3 Locating Placement Within the Grids

To test if the system was able to accurately locate where the game pieces were placed on the grid, multiple game pieces of altering sizes were placed in each one. The "ship" game pieces were first placed in the players' placement-grids and following the 1x1 attack pieces on the attack-grids.

Placement-Grid

For testing the accuracy within this grid, two different approaches were used. In the first approach the game pieces were placed randomly on the board, in a manner which is intended for the game. The second approach had odd placements, in order to see how the system would e.g. correct pieces with a crooked placement. An example of these approaches can be seen on Figure 6.2.



(a) *An example of the game pieces placed, as intended, on the board.*

(b) *An example of the game pieces placed with odd placements.*

Figure 6.2: *Images illustrating the two approaches.*

The test was conducted by having eight cases in total. In each case the pieces were placed, followed by a picture being taken. Afterwards the pieces were removed to show how the program projected them and additional pictures were taken of the outcome. The first six cases were for the intended placement and the last two were for the odd placements.

Attack-Grid

For this test there were two cases, those being one case for each player-grid. For the test, 13 mines were placed on each grid, whereas in the game, the player is only allowed to place four. This was to test a bigger ratio of the grid at a time.

6.2.4 Visual Feedback

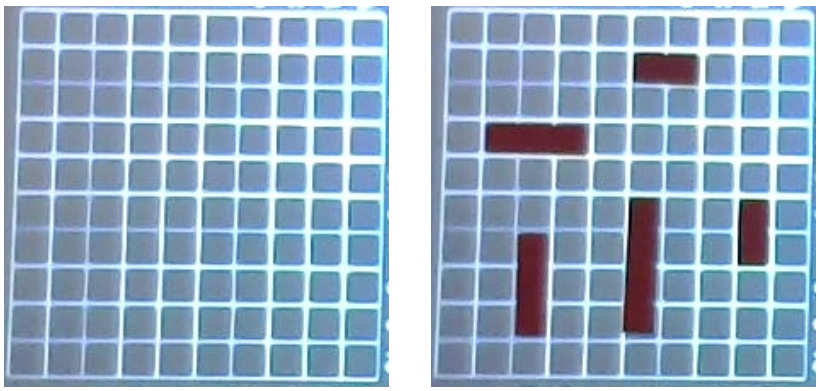
The visual feedback was tested both within the placement-grids and the attack-grids. Here, game pieces would be placed in each grid and afterwards removed, when the system had processed the data. After removing the pieces, should a visual representation (depending on the grid) be projected in its place. Additionally, visual feedback corresponding to the player's attack, should appear on the opponents placement-grid.

6.3 Results/Discussion

This section will go through the results of the tests, as well as discussions of the results.

Starting with *grid detection*, they were, as mentioned in Section 5.2.2, "detected"

through a "calibration" phase. Through testing the program, it was found that this generally worked consistently, when most of the set-up components remained still. Figure 6.3 shows a result of a cropped image, which the system saves throughout the game. Figure **a** is an image of one of the four cropped grids, before any game pieces are placed on them. Figure **b** is the same cropped grid, but with game pieces placed on it.



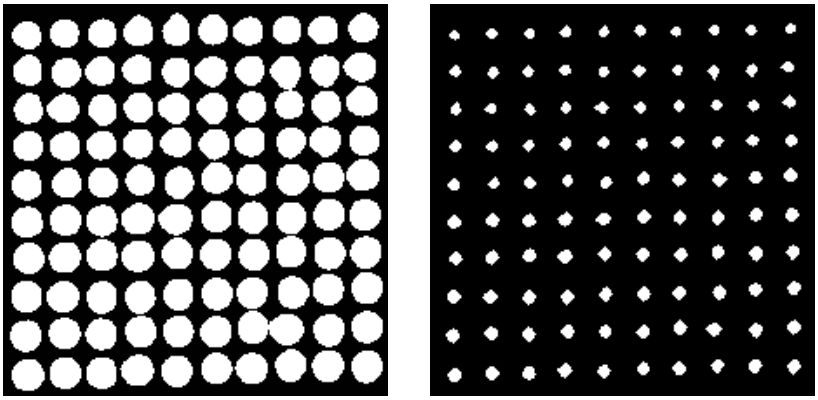
(a) *Figure of a cropped grid, without any game pieces.*

(b) *Figure of a cropped grid, with game pieces.*

Figure 6.3: *Images of one of the four cropped grids.*

This solution, in combination with the set-up, is very faulty in itself, which the two additional tests showed clearly. This is since, the cropping of the image only happens once before the game starts, meaning that if the projector or camera is moved during the game, the system will no longer bare able to accurately locate the game pieces within the grid.

The results from testing the *detection of game pieces* varied depending on the test. By using the 1x1 game piece, it was found that the game pieces were detected consistently, when there were not any shadows cast on the table surface. Figure 6.4 illustrates the detected BLOBs on each cell location in a grid, merged into one image. Additionally, the figure shows the eroded BLOBs.



(a) *An example of game piece placement in a single grid.* **(b)** *The eroded BLOBs from the example.*

Figure 6.4: *An example of the eroded BLOBs from the detected game piece.*

Since only one image was used for the background subtraction, instead of using the median of multiple images, it meant that any shadows cast on the table was considered a difference from the original image. An example, of how this affected the system can be seen on Figure 6.5.

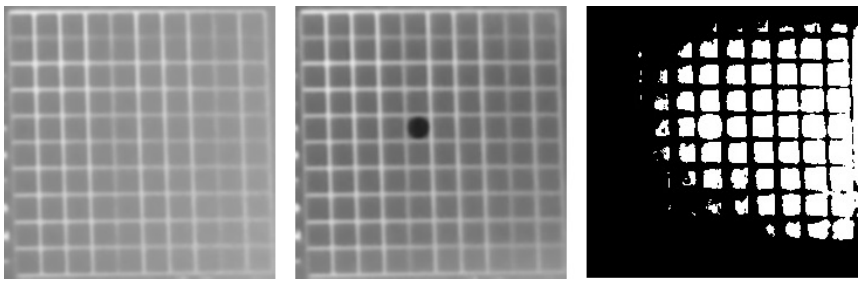


Figure 6.5: *Shadow impact on 1x1 game piece detection. The second grey-scaled image has a slight shadow, which impacts the detection after using background subtraction.*

When testing how accurately the system *located* the game pieces in the grid, it was found that it worked consistently on player 1's table-half. However, on player 2's table-half it had consistent issues with properly interpreting some game pieces placed vertically in the placement-grid. This became visible, when the projection of the ships appeared. By looking at the data stored in the corresponding array, it was found that the system would interpret some pieces as being bigger than they physically were, which interfered with how the system projected the pieces within the grid. An example of this can be seen on Figure 6.6. When looking at the binary- and eroded image of the BLOBs, the pieces were very similar to the physical ones, in contrary to the feedback which was projected. This can, furthermore, be seen on Figure 6.6. This means that the issue was not a projection error, but rather an

error in how the program processed its interpretation of what was physically on the table. This error was not occurring for ships placed horizontally, nor was it occurring on player 2's attack-grid.

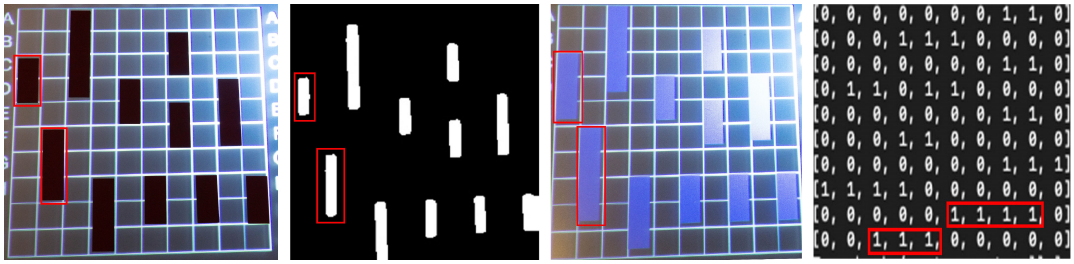


Figure 6.6: *Images illustrating the detection issue at player 2's half of the interactive table. The first image illustrates an example of placements within the player's placement-grid, the second the eroded BLOBs, the third the feedback projections corresponding to the placements and the fourth the data stored in the array corresponding to the specific grid. The data in the array is flipped 90 degrees counter clockwise.*

As mentioned, this was tested through different cases - one with proper placements and one with "odd" placements. Within the second case, where the odd placements were tested, multiple issues occurred. However, many of these were anticipated. The majority of the issues occurred, due to the way that the 10x10 grids are divided, as explained in Section 5.2.2. One issue was that if the game pieces were placed linearly in the grid, but slightly over the boarder of the intended axis and into a second adjacent axis, the system would e.g. interpret the game piece to be placed within the axis to the *left*. An example of this can be seen marked in red on Figure 6.7 and 6.8. However, if the piece crossed multiple boarders or had a crooked placement, it would attempt to correct it by placing the piece within the cells, which it was mostly lying within. This can additionally be seen on Figure 6.7 and 6.8, marked in green.

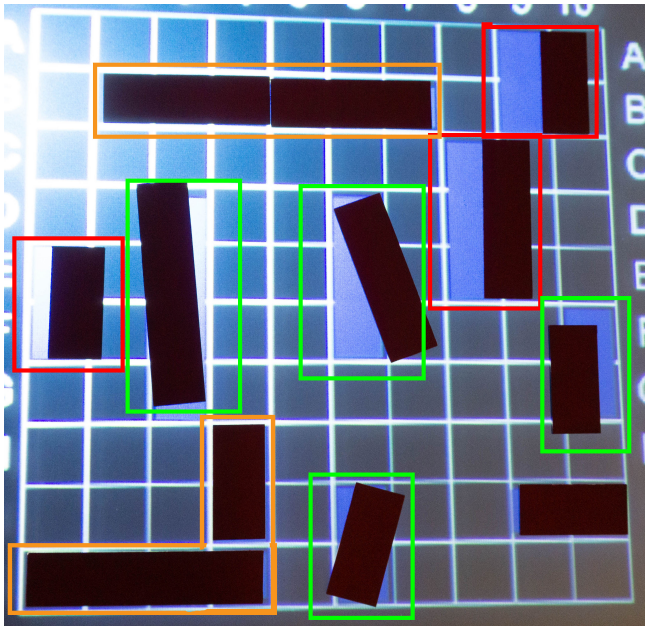


Figure 6.7: *An example of the results from the test with the oddly placed game pieces. The example illustrates the projection of the game pieces, with the game pieces still on the board.*

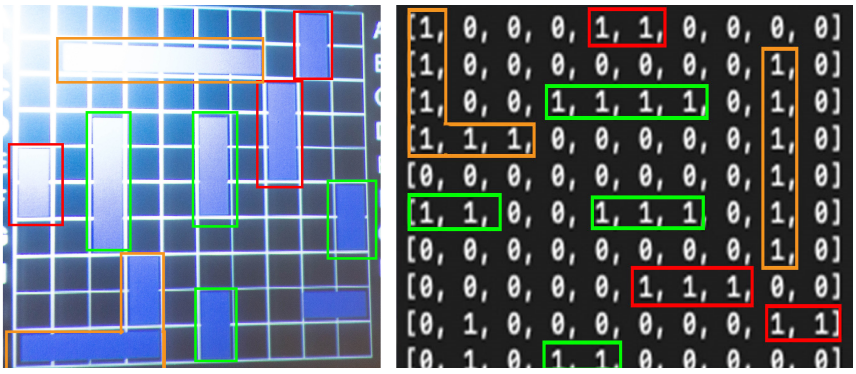
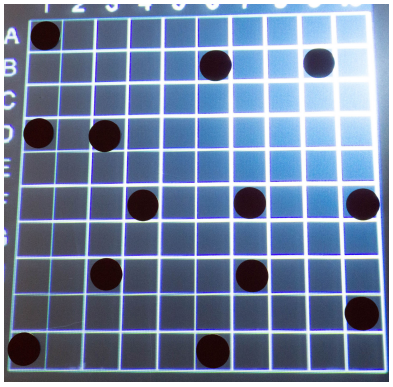


Figure 6.8: *An example of the results from the test with the oddly placed game pieces. The example illustrates the projection of the game pieces with the physical pieces removed and the data stored in the array corresponding to the grid. The array is flipped 90 degrees.*

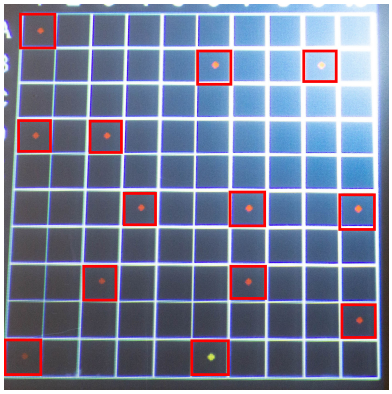
Lastly, for the odd placements, it was tested how the programme interpreted ships placed directly next to each other. The system was able to interpret and display the game pieces as separate ones, if there was a distance between the them. However, if the pieces had no distance between them, the system would interpret it as a single piece. This can also be seen on Figure 6.7 and 6.8, marked in orange.

These tests also gave insight into the accuracy of the *visual feedback*. Looking at the ship projections alone, they were consistently projected based on the data

stored within the arrays. However, if any shadows had interfered during the "calibration" phase, these would also interfere with the visual feedback. The projected visual feedback for the attacks was also found to work consistently, if not affected by shadows. An example from the tests can be seen on Figure 6.9.



(a) *An example of game piece placement within the attack-grid.*



(b) *The projected visual feedback.*

Figure 6.9: *An example of the eroded BLOBs from the detected game piece.*

Had a shadow, however, been captured on a player's attack-grid, would this be projected as if the player had attacked multiple places (by the attack piece itself, and from the shadow captured on the image). An example of this can be seen on Figure 6.10. Furthermore, if a shadow had been cast on the placement-grid, would the system either not know how to interpret the BLOBs and crash or project them as a cluster of ships.

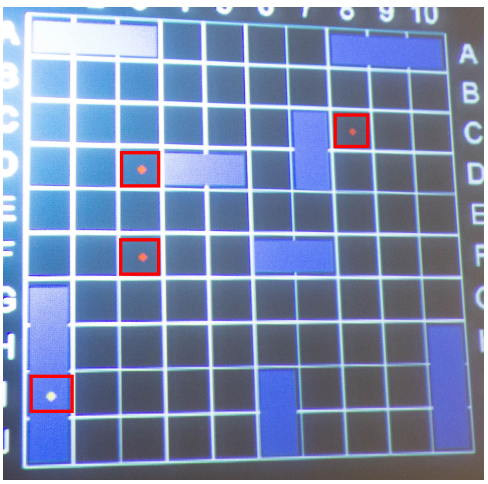


Figure 6.10: *Illustration of shadows impacting the game. The feedback (the red and yellow dots) came from the system interpreting a shadow as multiple attacks.*

Chapter 7

Discussion

Based on the results from the testing and evaluation chapter 6 it is clear that the prototype can be improved for a future iteration.

7.1 Evaluation

The detection of the created game pieces were not implemented as originally intended. Instead of the game pieces being classified based on shape and colour, any game piece despite shape and colour can be used as a ship or attack piece. This allows the users not to rely on certain game pieces in order to play the game. Furthermore the usability of the system have not yet been tested and it is therefore uncertain, whether or not the system is able to teach pupils in the age group from 12-19 years history. In order to test this the system would have to function as described in the design chapter.

7.1.1 Testing

The procedure for testing could be greatly improved, due to lack of discussion and planning beforehand, which caused extra work such as sorting images for results, which could have been avoided. This could have been achieved with a consistent documentation and evaluation plan. There have been many tests during implementation of various methods. However, the large majority of these tests were not documented due to the focus on the results, which led to inconsistencies in the evaluation which decreased the reliability of the tests, which meant that it would be difficult to replicate the tests in the exact same way. The lack of overview with evaluation meant that there was not a broad perspective which could have led to the use of more efficient methods, and could have sped up the process of implementation.

7.2 Further Development

Before considering additional development ideas for the system, the first goal would be to implement the described requirements that were intended, but were not implemented in this iteration. The system in its current state, is only running as intended on macOS and Linux systems because of system specific function parameters. In a future iteration another version of the system could be implemented to function with windows as well, by making a condition, that checks for which operating system the program is executed on, and use different function parameters based on that. Furthermore requirements such as implementing all the intended power-ups and projecting real battleships instead of coloured squares and rectangles simulating their position would also have to be implemented. However, due to this being a proof of concept, implementing all the mentioned power-ups and ship projections, is not necessary to fulfil the success criteria. Furthermore does the program require some preprocessing phases before the game is playable and these are gone through by the press of a button. In a future version of the game the preprocessing should either be done automatically or by an action detected by image processing. Additionally since the usability of the system have not yet been tested it is unsure whether or not the system fulfils its purpose of teaching pupils from the age group 12 to 19 years old history. In a final version of the system this would have to be tested and in case the history aspect is not fulfilling its purpose, it would have to be revised and adjusted.

7.2.1 Integration of History

The prototype is a proof of concept, and in the current state it does not include a satisfactory level of historical information. However, for further development, the current boxes should show different information throughout the game, and the ships should be displayed and named after real ships from the battle. With the game set-up it is not limited to teach history of the Battle of Leyte Gulf, or even World War two. The game has the potential to teach history about any naval battle in history, with different ship projection, facts, and tweaking of power-ups to fit with other historical settings e.g. If another iteration took place during a pirate battle set in the 18-hundreds, the ships would be projected as wooden vessels, and power-ups such as the aircraft carrier 3.4.4 would not be relevant. However it could be replaced with something like boarding enemy ships, where the largest ship would win, regardless there are many possibilities.
The motivation of using this interactive table might be an issue. The motivation for playing the game might not mean it is motivating to learn history per se, but that it is more about having fun with the game. However, it might not have an effect on the history the users will learn passively while playing the game, which is something that has to be tested for in future iterations.

7.2.2 Feedback

The visual feedback should be based on history, where projected ships should look like their historical counterparts from a top-view. However the size of the ships would be historically inaccurate. However, this could be notified in a short disclaimer that the ship sizes in the game will not be accurately represented. Furthermore the statistics of the ships should be displayed, e.g. the sizes and weight. The feedback could show realistic damage to the hull of the ships when they are hit, which could be done in various ways, such as displaying an explosion or smoke emitting from the ship. Misses could be displayed as splashes in the water, though the only import factor for visual feedback is for the grid-square to be marked throughout the entire duration of the game.
The current turn of a player must be displayed as visual feedback and changed relative to the turns, so that the players would always be able to keep track of which players turn it is. This would be done through the information boxes, displayed on the table, as seen on Figure 7.1.

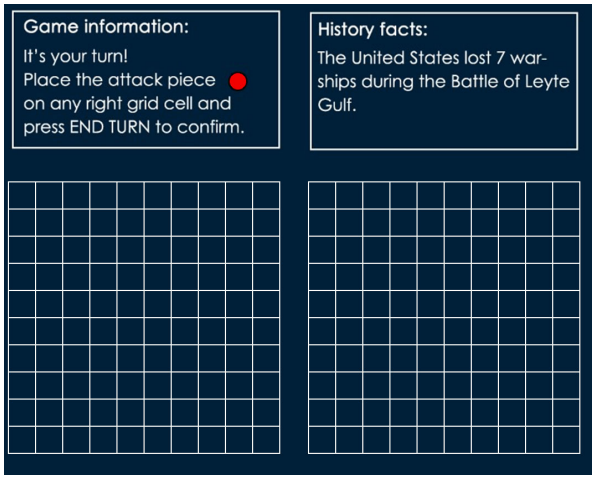


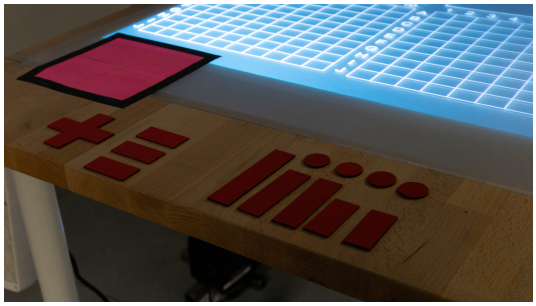
Figure 7.1: *Image of the visual feedback in the top left information box, keeping track of which players turn it is.*

Audio feedback could be implemented as well, e.g. when the player is attacking. The the sounds could be based around the used attack such as using the sound of a cannon fire, if a cannon power-up was to be used.

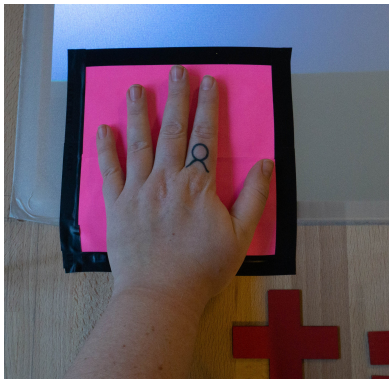
7.2.3 Hand Gestures

For ending turns, it was planned to use image processing, to eliminate the need for additional game pieces too keep track of. The idea was for the players to be able to end their turn with putting their hand on a pink square on the table as seen

on Figure 7.2. However due to technical issues during testing, this feature was not implemented in the final product. The details will be further discussed.



(a) *Set up from one side. Pink square for hand detection (left) game pieces (right).*



(b) *Example of hand placement for ending the turn.*

Figure 7.2: *Background subtraction with hand detection.*

Hand gesture consisted of four different processes. The first one being capturing images for background subtraction 5.1.2. Unlike the background subtraction used for attacking and placing ships, this had to be able to capture images from a live camera feed that runs continuously throughout the game, so it always would be able to detect changes. This was done by making a loop, where the camera would capture a single image every time a variable "t" reached 10 i.e. every time the program runs through the loop, it increments "t" by one and after running through 10 times, the program would capture the image and reset "t" to zero. Then this single still frame would be used as the background, and compare that with the ongoing live feed, with the same method as used in background subtraction 5.1.2. The reason for this was to make sure that the hand would be detected, even if there would be changes in the background, such as moving the table. The combined image is then converted to binary, where the background is black and the foreground is white as seen on Figure 7.3a.

After this segmentation process, the program should be able to detect a hand with using contours. After the image has turned binary, only the foreground should be visible, and in the best scenario, only the hand would be visible in the image. This hand would work as a BLOB, and from the that, the computer should be able to recognise it as a hand. To do this, the perimeter of the BLOB is measured, by using the length of the contours, which are the edges around the BLOB. This is done by summing the number of pixels in the contours. To implement contour detection, OpenCV [19] was used.

It extracts the contours from a binary image, and from here it stores each contour

(a) *Binary image after subtraction.* **(b)** *Image after grayscale and blurring.*

Figure 7.3: *Background subtraction with hand detection.*

as a vector of points. The amount of vectors are equal to the amount of contours. An issue could be that it would recognise other shapes than a hand, and to counter this, we calculate the contour area with OpenCV [19] by summing the amount of white pixels in the contours, and from here a threshold was set to filter out any unwanted BLOBs, if they were larger or smaller than a hand would be. After this, to specifically look for a hand and not any other object of that size, it would only look for BLOBs with more than 20 contours. However an issue with this could be that if another BLOB which is not a hand and has the same size and complexity or larger, the program would also recognise it as a hand. For the most parts this was not an issue, however sometimes it would detect shadows as a hand if people walked around the table, while the game was played. The instances of this was decreased by blurring the image even more, however the issue would still arise.

The code for hand detection was not implemented due to complications with merging it with the main module. The main module used a “key pressed” as an input i.e. the player press and hold “E” on the keyboard, and it would return a “True” which let the program know to move on to another module where the turn would end and switch to the other player. However, when implementing the hand gesture, which would return a “True” or “False”, the program would never enter the background subtraction module, which caused the program to crash.

7.2.4 Adaptive Calibration

From the results of the grid detection as described in Section 6.3, it was concluded that moving the projector had a great effect on detection, which meant that in order to play the game properly, it should stay stationary throughout the game, otherwise the game had to be reset. This could have been improved by making the

calibration, as mentioned in Section 5.2, adaptive. Instead of using a still image in the beginning of the program and finding the coordinates from there, it could use a live video feed, as described in Section 7.2.3 about hand gestures. When the program crops the grid in the beginning, there could instead be a timer, where the grid would be re-cropped in a time interval i.e. if the set-up was moved throughout the game, the grids would automatically be re-cropped and moved, which means that the game would still be playable. To further increase the accuracy of background picture 5.1.2, instead of capturing a single image at a time intervals, multiple images could be captured at the time, where the median of the combined images could be used as a background reference, to decrease the effect of unwanted objects in the image, such as shadows.

7.2.5 Detection

As mentioned in the evaluation the detection of game pieces proved to be somewhat consistent. When looking at player one the detection of mines, ships and attacks proved to work every time, however, for player two only the mines and attacks was consistent. It was found that when placing ships vertically on the right side of the placement grid, the program would portray the ship size to be one size bigger therefore, projecting a too large ship. However, it still stores the data correctly the ship is just too big. The game is still playable and it does not affect the gameplay in itself, but it gives player two an unfair advantage since they attain bigger ships then intended.

Furthermore, another issue that arose with detection was that if two ships were placed on adjacent locations, very close to each other, to the point of touching against one another, the program might detect them as being one huge ships. This would cause problems with representing the ship properly in an array. A fix for this would be a gameplay limitation, which would forbid the players from placing two ships adjacent to each other, by making it a rule.

As mentioned in Section 5.2.2 the game currently finds the pieces by using background subtraction and mathematical logic, however, as seen in evaluation it is inconsistent. A way to solve this could be by using more than one background subtraction image pr. grid and take multiple as mentioned in Section 7.2.3, this would increase the consistency of the subtraction and furthermore, decrease the risk of light and shadows illuminating the grid.

7.3 Wider Context

Not only would the current system be able to be further adapted, the current setup also has the potential of being used in other settings, both as an educational tool and as a tool for entertainment.

Furthermore, a discussion about about how the current system would affect both the teachers and pupils has to be had. Firstly, the system might work well as a supplementary educational tool for teachers. They could enhance their lecture by letting the pupils play in the historical setting they are learning about. This might make it easier for teachers to capture the attention of pupils and make them more interested in the class. On the downside, the teachers might lose the attention of the children due to the interactive table taking up too much focus. Furthermore, since this would be an unconventional way to teach, it might be hard for it to be accepted by more traditional educational systems, which would make it more difficult for teachers in those countries or regions to use it to its full potential. Also, since it is not a technology that is widespread, some teachers that are not well acquainted with using the elements of the system might have problems with setting up the system.

From the pupils side of things, the system might be able to capture their attention more easily. It could also help them feel more immersed in the history they are being taught. If they find the game exciting, it might also serve as an incentive to them to pay more attention in class in order to find out more about the battle or conflict they played.

Another stakeholder in this would be the educational institution itself. As mentioned previously, it is unlikely this system would be accepted by all educational systems. Besides this problem, there is also the trouble that the institution would have to pay for each element of the system themselves. This might potentially mean a projector and a dedicated laptop for each system they would use. However, some institutions already have projectors, meaning cost could be reduced in that regard. Also, the elements which make up the system would have to be well taken care of which might also add another inconvenience to the school. However, if an institution is willing to look past these drawback, the interactive table might prove to be a good addition to the lectures, and an alternative way of teaching which would captivate students more then the traditional way. To make the system more user friendly regarding set-up and components, the system has to be made more adaptable.

However it would also be possible to implement existing games, design new games

or other interactive systems using the same setup. Traditional board game designs such as Monopoly and scrabble could be implemented in similar fashion as the presented prototype. Furthermore by implementing existing board games the need of a physical copy would be eliminated since a version could be stored on a computer.

Chapter 8

Conclusion

Based on the evaluation of the implementation it can be concluded that the solution in its current state is working as intended with the majority of the implemented features, however it is still lacking features such as using hand gesture to end a turn, the remaining power-ups and history teaching elements, to fulfil its purpose of being able to teach 12–19 year old pupils history. Furthermore, the implementation was set to be modular, however, it ended up not being due to every module being dependent on another one. Additionally, looking at the modules themselves they can be improved greatly upon due to the inconsistency they showed to have. This can be seen in both the calibration and detection module, where the threshold is global and not adaptive. Furthermore, the background subtraction only having 1 reference picture pr. grid. This can however be solved by implementing an adaptive threshold that would change dynamically based on the pixel it is on, but also using more reference pictures for the background subtraction e.g., by taking the mean of multiple images to create a more valid reference picture. Also the documentation for the implementation iterations was not well conducted which led to inconsistent evidence on how the solution was created and furthermore, which methods that has been tried throughout the project. Lastly, the proof of concept revolving around history was never tested nor implemented and therefore could not be concluded, thus ending the project at a point in time where it is uncertain whether it was a success or not, however, in a future iteration the solution could be improved greatly upon by implementing the remaining features that were intended to be implemented, however as they were not, it could not be tested whether the final problem was solved or not.

Bibliography

[1] *Axis and Allies image*. URL: http://www.digitallydownloaded.net/2019/02/classic-board-game-axis-allies-heads-to.html.

[2] *Battleship*. 1990. URL: https://www.hasbro.com/common/instruct/Battleship.PDF.

[3] Steve Benford, Carsten Magerkurth, and Peter Ljungstrand. "Bridging the physical and digital in pervasive gaming". In: *Communications of the ACM* 48.3 (Mar. 2005), p. 54. ISSN: 00010782. URL: http://portal.acm.org/citation.cfm?doid=1047671.1047704 (visited on 09/27/2019).

[4] *BoardGameGeek | Gaming Unplugged Since 2000*. URL: https://boardgamegeek.com/ (visited on 10/10/2019).

[5] John Brooke. "SUS: A Retrospective". In: ().

[6] Bill Buxton et al. *Sketching User Interfaces, the Workbook*. 2012, p. 262. ISBN: 9780123819598. URL: http://www.amazon.de/Sketching-User-Experiences-Bill-Buxton/dp/0123819598/ref=pd{_}sim{_}eb{_}18.

[7] Emile J.L. Chappin, Xanna Bijvoet, and Alexander Oei. "Teaching sustainability to a broad audience through an entertainment game – The effect of Catan: Oil Springs". In: *Journal of Cleaner Production* 156 (July 2017), pp. 556–568. ISSN: 09596526. URL: https://linkinghub.elsevier.com/retrieve/pii/S0959652617307874 (visited on 09/27/2019).

[8] *Chekered Game of Life image*. URL: https://www.design-is-fine.org/post/155245854879/the-checkered-game-of-life-1866-milton.

[9] Richard E. Clark. "Learning from Serious Games? Arguments, Evidence, and Research Suggestions". en. In: 47.3 (2007), pp. 56–59. URL: https://www.jstor.org/stable/44429512 (visited on 09/27/2019).

[10] Mary Flanagan. *Critical Play: Radical Game Design*. Cambridge, UNITED STATES: MIT Press, 2009. ISBN: 978-0-262-25876-0. URL: http://ebookcentral.proquest.com/lib/aalborguniv-ebooks/detail.action?docID=3339056 (visited on 09/27/2019).

[11] *geo-game 2 Battleship image*. URL: https://www.researchgate.net/figure/Battleships-online-game-screen-source-http-migreme-iM49l_fig1_277881613.

[12] *Hasbro Battleship image*. URL: https://lekmer.dk/hasbro-gaming-battleship-se/p/141449.

[13] Jessica Ip and Jeremy Cooperstock. "To Virtualize or Not? The Importance of Physical and Virtual Components in Augmented Reality Board Games". en. In: *Entertainment Computing – ICEC 2011*. Ed. by David Hutchison et al. Vol. 6972. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 452–455. ISBN: 978-3-642-24499-5 978-3-642-24500-8. DOI: 10.1007/978-3-642-24500-8_64. URL: http://link.springer.com/10.1007/978-3-642-24500-8_64 (visited on 09/27/2019).

[14] *KnightMage image*. URL: https://www.researchgate.net/figure/The-STARS-tabletop-platform_fig3_220686485.

[15] Mikhail Kurushkin and Maria Mikhaylenko. "Orbital Battleship: A Guessing Game to Reinforce Atomic Structure". In: *Journal of Chemical Education* 93.9 (Sept. 2016), pp. 1595–1598. ISSN: 0021-9584, 1938-1328. URL: https://pubs.acs.org/doi/10.1021/acs.jchemed.6b00136 (visited on 09/27/2019).

[16] *Mansion of Happiness image*. URL: https://en.wikipedia.org/wiki/The_Mansion_of_Happiness.

[17] Francesco Bellotti Giusy Fiucci Minica Houry-Panchetti et al.. Michela Mortara Chiara Eva Catalano. *Learning cultural heritage by serious games | Elsevier Enhanced Reader*. en. 2015. URL: https://reader.elsevier.com/reader/sd/pii/S1296207413001349?token=B843376FE315627FF43B2D60BF4B456E36472D3518C0D6653C06 (visited on 09/27/2019).

[18] Thomas B. Moeslund. *Introduction to video and image processing: building real systems and applications*. en. Undergraduate topics in computer science. OCLC: 839022745. London: Springer, 2012. ISBN: 978-1-4471-2502-0 978-1-4471-2503-7.

[19] *OpenCV*. URL: https://opencv.org/about/ (visited on 10/22/2019).

[20] Jenny Preece, Helen Sharp, and Yvonne Rogers. *Interaction Design: Beyond Human-Computer Interaction*. Engelsk. 4th ed. Vol. 2015.

[21] Marc Prensky. *Digital game-based learning*. en. New York: McGraw-Hill, 2001. ISBN: 978-0-07-136344-0.

[22] Fran Salyers and Carol McKee. "The Young Adolescent Learner". en. In: (2007).

[23] Karen Schrier and International Game Developers Association, eds. *Learning, education and games*. en. OCLC: ocn905668700. Pittsburgh, PA: ETC Press, 2014. ISBN: 978-1-312-54285-3.

[24] *Shining Rock Software*. en-US. URL: http://www.shiningrocksoftware.com/ (visited on 12/19/2019).

[25] Kurt Squire and Sasha Barab. "Replaying History: Engaging Urban Underserved Students in Learning World History Through Computer Simulation Games". en. In: (), p. 8.

[26] Kurt Squire and Sasha Barab. "Replaying History: Engaging Urban Underserved Students in Learning World History Through Computer Simulation Games". en. In: (), p. 8.

[27] *The Catan: Oil Springs Scenario image*. URL: https://i.redd.it/h3ayt3tnxxy01.jpg.

[28] *The Grizzled image*. URL: http://brokenmeeple.blogspot.com/2015/10/code-name-operation-certain-death.html.

[29] "THE "GYMNASIUM" – A DANISH UPPER SECONDARY EDUCATION". en. In: 20 (Nov. 2013), pp. 4–15. URL: http://www.danskegymnasier.dk/wp-content/uploads/2014/11/Engelsk-pjece-til-web.pdf (visited on 10/11/2019).

[30] *Time spent gaming weekly among children UK 2018*. en. URL: https://www.statista.com/statistics/274434/time-spent-gaming-weekly-among-children-in-the-uk-by-age/ (visited on 12/13/2019).

[31] *TotalwarGames*. en-GB. URL: https://www.totalwar.com/games/ (visited on 10/07/2019).

[32] *What do the labels mean? | Pegi Public Site*. URL: https://pegi.info/what-do-the-labels-mean (visited on 10/11/2019).

[33] José P. Zagal, Jochen Rick, and Idris Hsi. "Collaborative games: Lessons learned from board games". en. In: *Simulation & Gaming* 37.1 (Mar. 2006), pp. 24–40. ISSN: 1046-8781, 1552-826X. DOI: 10.1177/1046878105282279. URL: http://journals.sagepub.com/doi/10.1177/1046878105282279 (visited on 09/27/2019).

Appendix A

Lo-fi script

Process:

- Explain the purpose of evaluation
- Sign the consent form and explain what type of
- data we will derive from the evaluation
- Give them instructions
- Read instructions on the table
- Ask to make sure they understand it
- Start the game
- Place their ships
- Place brick where they want to shoot
- End your turn
- Go back to step 2 until the game is over
- Answer User Experience Questionnaire
- Answer additional questions

Additional questions

1. Did you notice anything changing throughout the game?
2. How intuitive did you find the game?
3. What are your thoughts about the layout

4. What do you think about the colour scheme?
5. Any thoughts about visual and audio feedback?
6. Any other comments?

Manuscript:
You are participating in a lo-fi evaluation, for testing the functionality and design of an interactive table game. with the purpose of teaching history.

This test is only focusing on the design aspect of the game, and not whether or not you as the participant learn any historical facts.

Throughout the test, you will be audio and video recorded for the purpose of gathering as much data as possible for further analysis, therefore you have to sign a consent form from GDPR.

(Sign the consent form)

Your goal is to play the game of battleships presented in front of you and think aloud while you are doing it. Here you can say what you want about everything or ask questions if you are in doubt of anything. You will be playing for 10 minutes, where afterwards we have a SUS questionnaire and additional questions we would like you to answer.

(After the participants have answered the questions)

Thank you for participating in our evaluation.

Appendix B

Lo-fi Interviews

An example of the Lo-fi interviews can be found below. The rest of the interviews can be found in the C folder.

1st. Participant lo-fi Interview
I = Interviewer
P = Participant

I: alright, so we have a few questions we would like you to answer and then we have a questionnaire. So, did you notice anything changing throughout the game?

P: uuuh.. almost at the end, I noticed that the.. the information - the history information thing changed.. though I didn't really read it. Only once because I saw it changed.

I: Mm-hm. So, you didn't - did you feel like you had time to read it or you wanted to read it?

P: I had the time, I just.. didn't do it.

I: Is there a particular reason for why you didn't?

P: no, I was too.. caught up in the game.

I: caught up in the game...

P: yeah..

I: that's good, heh. So what do you think about putting the game pieces on the

board and then taking them off again?

P: I think it's.. I like it, it's like... some sort of calculat- calibration thing.. so.. you do something physical where you're putting pieces on it and take them off.

I: Okay. So it doesn't bother you?

P: No. I guess it's just easier to.. like.. calibrate.

I: Right.. So how intuitive did you find the game?

P: Very intuitive.

I: Very intuitive?

P: Yeah, the.. the button thing that you had to push every time you finished.. yeah, heh

I: nice. So what are your thoughts about the layout?

P: I liked the layout. Eeeh.. but I hope.. - I know that this is lo-fi, but.. you are going to change it so they can be like.. horisontal too..

I: yeah

P: Yeah, good.. good

I: heh, so anything about the layout you didn't like? The placement, different things?

P: No.. I think it looked nice.

I: It looked nice..

P: Yeah. I like that it's simple.. nothing too much..

I: Yep.. any thoughts about the color-scheme?

P: I mean.. it makes sense that "hit" is red.. and.. yeah.. yeah, I like red and blue, heh

I: Alright.. red and blue is great. So anything about the grid? The white grid and the dark background? Any thought about that?

P: Mmmh, no it's.. but it's easy on the eye.

I: It's easy on the eye..

P: yeah

I: Right. Any thought about visual and audio feedback?

P: I mean.. The- the thing- the noise- the sound that.. that the button makes.. it sounds like it's a miss every time, because it says the same thing.

I: yeah

P: eeh.. I think it would be nice to change.. that.. because it sounds like it's a sound that would.. be there when it's a miss

I: Yeah, it's also.. I also kind of.. kind of criticized it, because I think it sounds like when you're on a gameshow.. and you get the answer wrong-

P: Yeah, exactly, it's like the buzzer thing-

I: Like a "BWAA"-

P: Yeah, heh, exactly..

I: Do you have any other comments?

P: Eeeh.. no.. I think it's a nice lo-fi. You did a good job.

I: Thank you.

Appendix C

Full Appendix

The Full Appendix is attached as a separate folder and consists of the following material:

- AV production
- Lo-fi interviews
- Consent form template