UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

INCA-CLICK IN ORDER: APLICACIÓN WEB PARA ESTUDIAR LA MEMORIA DE TRABAJO EN UNA Y DOS DIMENSIONES DE COTORRAS ARGENTINAS

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN COMPUTACIÓN

ABRAHAM OQUICHE AVILA

PROFESOR GUÍA:
JÉRÉMY BARBAY

MIEMBROS DE LA COMISIÓN:
SERGIO OCHOA DELORENZI
JUAN ÁLVAREZ RUBIO
JOHAN FABRY

SANTIAGO DE CHILE
2024

Resumen

Mientras que Inoue y Matsuzawa [3] describieron en 2007 cómo algunos chimpancés mostraron una mejor memoria de trabajo para permutaciones bidimensionales que algunos sujetos humanos, Silberberg y Kearns [5] mostraron en 2009 que cuando dos humanos reciben suficiente práctica, sus niveles de precisión igualan a los del mejor chimpancé. El software (proprietario) para ambos estudios, que supone el aprendizaje preliminar de la secuencia de números arábigos (actualmente utilizados en la mayoría de lenguas occidentales) del 0 al 9, solicita a los sujetos memorizar una secuencia de ubicaciones bidimensionales en una pantalla en un tiempo limitado.

La tarea de enseñar a los sujetos aumenta el costo de reproducir sus experimentos con sujetos de especies distintas a los humanos o con humanos jóvenes. ¿Puede reducirse el costo de reproducir y ampliar tales estudios (por ejemplo, a especies distintas de chimpancés y humanos, y a humanos más jóvenes) mediante el uso de otras representaciones numéricas más naturales, como una nube de n puntos para representar el número n? Diseñamos, desarrollamos y validamos una solución web simple para replicar y ampliar el software utilizado en ambos estudios, que puede configurarse para utilizar tanto las mismas representaciones numéricas que en sus estudios, como otras representaciones numéricas que podrían considerarse más naturales, como una nube de puntos o círculos de tamaños distintos.

*A mi familia y amigos que, a pesar de la distancia,
me hicieron compañía y apoyaron mis sueños.*

Agradecimientos

Al profesor Jérémy Barbay, quién me guió a lo largo de este proyecto, compartiendo su experiencia y conocimiento. Su compromiso con el tema de esta memoria y con la enseñanza me ayudó a crecer como profesional.

A Lorenzo y Tina, cotorras argentinas usuarias de la aplicación, que aunque se frustraron debido a errores míos durante el desarrollo, siguen disfrutando de `InCA-ClickInOrder`.

Table of Content

List of Tables

List of Figures

Chapter 1

Introduction

In this chapter, some background on the state of the art will be mentioned, along with the detected problem that this project aimed to solve. Then, the objectives needed to accomplish this are listed, and a solution that embraces such objectives is stated. Finally, the contents of this document are presented to give a general view of the matters written in this report.

1.1. Background

In 2007, Inoue and Matsuzawa [3] investigated the working memory of chimpanzees on two-dimensional permutations. They conducted a series of experiments on a group of six subjects: three mothers, and offspring pairs. With a touchscreen that showed the Arabic numerals from 1 to 9 in random positions among 40 different locations on the screen, the subjects were trained to select the correct ascending sequence of numerals. This showed how some chimpanzee subjects displayed a better working memory on two-dimensional permutations than some human subjects while playing a simple video game requesting them to memorize a sequence of two-dimensional locations on a screen in a limited amount of time.

They later described an extension of their experiments on chimpanzees [4] comparing the learning capabilities and memory capacity of sequential ordering between young and adult chimpanzees. This showed no difference regardless of age in the ability of sequential learning, however, the younger ones outperformed the adults in both accuracy and difficulty, regarding the amount of numerals achieved, when memorizing a sequence.

In 2009, Silberberg and Kearns [5] contested those results by recreating the experiments and showing that when two humans are given practice in the Inoue and Matsuzawa memory tasks, their accuracy levels match those of the best chimpanzee subject.

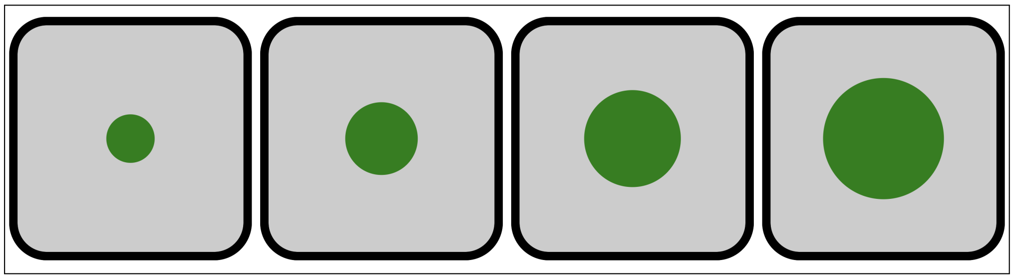


Figure 1.1: Numerical representations using a circle that grows in diameter.

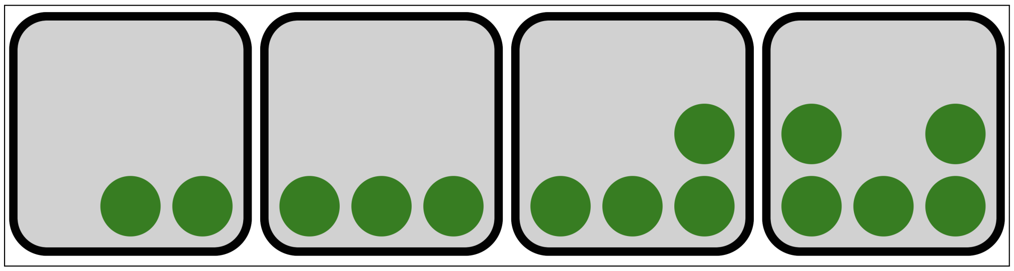


Figure 1.2: Numerical representations using a cloud of dots.

1.2. Problem

The (proprietary) software for both studies supposes the preliminary learning of the sequence of Arabic numerals (the ones currently used in most Western languages) from 0 to 9: teaching subjects is increasing the cost of reproducing their experiments with subjects from species other than humans or with humans of younger age. In a distinct experiment requesting the subjects to identify the largest of a small set of numerals, Barbay et al. [2] solved such an issue by using other, more natural, representations of a numeral n, such as a cloud of n dots or a circle of diameter proportional to n (see Figures 1.1 and 1.2 for such examples).

Can the cost of reproducing and extending (e.g. to species other than chimpanzees and humans, and to younger humans) the studies of Inoue and Matsuzawa [3] and of Silberberg and Kearns [5] be reduced by the use of such alternate numerical representations? Can this reduced cost allow researchers to perform new experiments in the hope of resolving the discrepancy between those two studies?

1.3. Objectives

As a means to provide a solution to the problem mentioned, a specific objective and specific outcomes are mandatory. These bullet points cover what the piece of software implemented needs to be capable of before reproducing the studies discussed in Chapter 2.

1.3.1. General

The general objective of the work was to design, develop and validate a software allowing to replicate and extend the two studies from Inoue and Matsuzawa [3] and Silberberg and Kearns [5], by using the alternating numerical representations described by Barbay et al. [2] in a way extendible to alternate representations. We decomposed this objective into separate outcomes in the next section.

1.3.2. Specific

In achieving the general objective, we produced the three following separate outcomes:

1. a front-end aimed at subjects previously trained in the use of touch-screens, allowing to replicate the two studies from Inoue and Matsuzawa [3] and Silberberg and Kearns [5] and to extend it with new numerical representations, while sending a detailed log of the subject's interaction to a remote database;

2. a server receiving and storing such detailed logs of interactions for future analysis, in a secure way. A front-end aimed at researchers was developed by Cristobal Sepulveda, allowing them to selectively analyze and/or download the logs of interaction in order to be able to validate or invalidate the various hypotheses from the two studies from Inoue and Matsuzawa [3] and Silberberg and Kearns [5], and eventually additional hypothesis; and

3. a validation of the usability of the front-end aimed at subjects with Quaker parrots.

1.4. Solution

We designed, developed, and validated a simple open-source web solution to replicate and extend the software used in both studies. This application can be configured to use either the same numerical representations as in their studies, or other numerical representations. The last ones could be considered more natural, such as a cloud of dots or circles of distinct sizes (see Figures 1.1 and 1.2), and such that additional numerical representations can easily be added, and the results of additional experiments can easily be compared to previous ones.

1.5. Contents

We describe in more detail the experimental setups and results from Inoue and Matsuzawa [3], Silberberg and Kearns [5] and Barbay et al. [2] in the next Chapter. Then, we list the software features of the solution and present its design along with the development methodologies used (in Chapter 3). Next, the implementation itself is shown and thoroughly described, with explanations of the decisions made during development (in Chapter 4). Afterwards, we talk about the process of validating the application and the outcomes (in Chapter 5) presenting not only an analysis of the data collected but the troubles that appeared during the process. At last, a summary of what was accomplished (in Chapter 6) and what was left in the backlog (in Chapter 7) is stated.

Chapter 2

Previous Studies

This work was inspired by three distinct studies, each using a separate piece of software. We describe those studies in the following Section 2.1 and give a conclusion related to the noticeable disagreement in Section 2.2.

2.1. Summaries of Previous Studies

The study from Inoue and Matsuzawa [3] describes an experiment on the working memory of chimpanzees and compares their results with the ones of human subjects, with the chimpanzees showing a better performance. Then the study from Silberberg and Kearns [5] suggests that, when given practice, the performance of both human and chimpanzee subjects match. Finally, the study study from Barbay et al. [2] proposes that adding digital enrichment techniques can reduce the cost of experiments and improve precision.

2.1.1. 2007 study from Inoue and Matsuzawa

As previously mentioned, Inoue and Matsuzawa [3] investigated the working memory of chimpanzees on two-dimensional permutations. Conducting a series of experiments on a group of six subjects: three mothers, and offspring pairs. The setup consisted of a touchscreen that showed the Arabic numerals from 1 to 9 in random positions among 40 different locations on the screen, and the subjects were previously trained to select the correct ascending sequence of numerals.

In the experiments, they introduced the masking task to the subjects. This tasks consists of replacing the numbers with white squares after the first numeral gets selected. After performing this test, the results showed that all the subjects mastered this task, even though the performance of the younger subjects was better than the one of their mothers.

Then, the limited hold memory task was introduced to the chimpanzees; this consisted of giving a certain amount of time before covering the numbers. Three different hold durations were tested: 650, 430, and 210 milliseconds. At this stage, the subjects' performance was compared with human subjects: a group of 9 university students.

Overall, the results showed that the young chimpanzees performed better than both adults and human subjects, especially the subject named Ayumu, who got approximately 80 % accuracy during the test with the three hold durations. In contrast, the human subjects got an average of 40 % accuracy at the 210 ms hold duration and the older chimpanzees an average of 20 % accuracy with the same hold duration. Inoue et al. [3] explained this by what is known about eidetic imagery in humans, which declines with age.

2.1.2. 2009 study from Silberberg and Kearns

Silberberg and Kearns [5] recreated the setup of the experiments held by Inoue et al. [3], aiming to show that when given enough practice, humans can achieve the performance shown by the young chimpanzees. Since the software used by Inoue and Matsuzawa was not made public, Silberberg and Kearns reimplemented the logic in a web application.

They first performed the masking task, where the numbers were replaced by white squares after the first numeral was selected, giving the subjects enough time to memorize the correct sequence. This task was held with a group of students, and the results showed that the 12 untrained human subjects performed better than any ape from Inoue and Matsuzawa's 2007 study [3]. For this reason, the second test focused on training to get the best results in the limited hold task.

This second task, which was carried out by the authors instead of a group student, consists of giving an amount of delay after the task begins before covering the numbers with a white square. Thus, the authors of this study started training, doing 50 trial sessions, varying the number of sessions per day from three up to ten, with different hold duration from 250 to 100 milliseconds. At the latency of 210 ms, the shortest one used in the 2007 study, both adults were able to match the accuracy level of Ayumu, the chimpanzee who performed the best in the limited hold task. To get this result the subjects took around 2500 trials throughout their sessions.

Inoue and Matsuzawa report [3] that humans got up to 40 % accuracy with the 210 milliseconds task, which is similar to the one achieved by the subjects before training, so Silberberg and Kearns [5] results suggest that when given enough practice, the performance of adult humans can match the one from young chimpanzees, contradicting the conclusions of Inoue et al. [3].

2.1.3. 2022 study from Barbay et al.

Barbay et al. [2] studied the ability of Quaker parrots to discriminate different quantities using both discrete and continuous formats. By developing an application called `InCA-WhatIsMore`, they aimed to reduce the cost and to solve potential issues with the previous study from Al Aïn et al. [1]. Adding digital life enrichment techniques allowed them to get more results while reducing the cost of the experiments, improving precision, and reducing the potential bias that could be inadvertently induced by the experimenter.

The software developed consists of a web application in which the representations are displayed in a single row and the subject has to choose the highest quantity shown on screen using different representations, like a dice or a disk. Thanks to the audio feedback, the researcher can set up the touchscreen in such a way that the subject is the only one aware of what is being shown. The data from the session is automatically collected using logs designed to be easily readable by humans.

During the experimentation phase, in which the values were restricted to the set {1, 2, 3, 4, 5, 6}, the two subjects had to choose the maximal value out of two, three, and four quantities. The results showed that for two values the two subjects got an accuracy of 82 % and 74 % respectively, similar to results described in previous studies with all the digital benefits already mentioned.

2.2. Summary

The apparent conflict between the results described by Inoue and Matsuzawa [3] on one hand, and by Silberberg and Kearns [5] on the other hand suggests the need for further studies on the working memory of humans and animals other than humans. The alternative numerical representations described by Barbay et al.'s study [2] suggest that such additional studies could be made with less preliminary training than previous ones, given the proper software. We describe the process of designing, implementing, and validating such software in the following chapters.

Chapter 3

Solution

This chapter extends the list of functionalities mentioned in the state-of-the-art studies from Chapter 2 and also explains how these features were portrayed in the application. Also, what was taken into consideration when designing the software developed is stated, with the methodology used during development.

3.1. Baseline

It is important to notice that the template used for this project was `InCA-WhatIsMore` and even though the code provided allowed me to get used to the technology and learn what approach was used for that application, a lot of improvements has to be done in order to get to the current solution presented in `InCA-ClickInOrder`. In that regard, the template had many restrictions that had to be reimplemented before adding the new features, some of which were:

- Allowing to have representations displayed only in a single row and not dynamically positioned.
- The local log format did not fit the requirements of having flexibility and an accurate description of the interactions.
- The way of saving the settings did not allow the users to keep their current configuration after a new version arrived.
- The game menu was static and the information of which mode was selected was not being collected.
- An overall difficult-to-use user interface.
- The way interactions were noticed by the application did not allow the new modes to be implemented. This is due to the logic not being encapsulated within components but a large script.

Thus, many parts of the given code had to be rethought and redesigned before adding the new features. Therefore, this project consisted of refactoring what was given using the tools and knowledge of a software engineer and implementing all the new features shown in Chapters 3 and 4.

3.2. Software Features

Among the features of the software described in the sections above, we distinguish in particular three types of features, respectively, related to the numerical representations being used (Section 3.2.1), to the order of the numerical sequences being considered (Section 3.2.2), and to how a log of the interactions of the software is kept for later analysis (Section 3.2.3). It's also important to notice that in the research from Inoue et al. [3], the size of the array was fixed and in `InCA-WhatIsMore` the representations were next to each other within a single row. The software developed, `InCA-ClickInOrder`, generalizes that behaviour allowing to specify a matrix size in the settings.

3.2.1. Numerical Representations

The three studies [2, 3, 5] previously described introduce a total of five numerical representations:

1. **Arabic:** This one uses the numbers in the set {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}.
2. **Dice:** A dice face is defined as a matrix of dimension 3x3, in which each position of the array can be filled with a spot. The dice representation consists of a fixed pattern for each number (see Figure 3.1).
3. **Heap:** Similar to the dice face defined earlier, in this representation's dots are arranged filling each row from left to right and bottom to top. The heap consists of a fixed pattern for each number where the spots follow the rule described (see Figure 3.2).
4. **Rectangle:** This continuous representation is similar to a recipient filled with liquid. Here, an empty square is used as a vessel, and a rectangle that grows from bottom to top represents an amount of liquid (see Figure 3.3).
5. **Disc:** Similar to the rectangle representation, the disc is a circle fixed in the center that grows in all directions surrounded by an empty square (see Figure 3.4).

While the Arabic representation was used in both studies by Inoue and Matsuzawa [3] and Silberberg and Kearns [5], the dice, heap, rectangle, and disc were used in the study by Barbay et al. [2]. Representations like the Arabic numbers serve the purpose of showing symbols commonly used by humans around the globe. On the other hand, other representations have a more natural approach, where each number tries to mimic seeds or liquid in the wild.

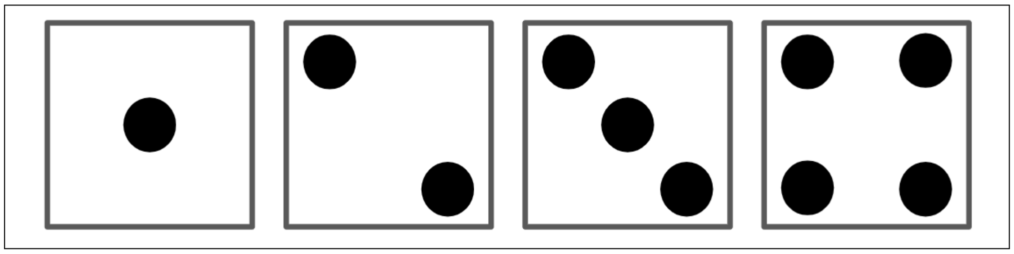


Figure 3.1: Numerical representations of the dice pattern.

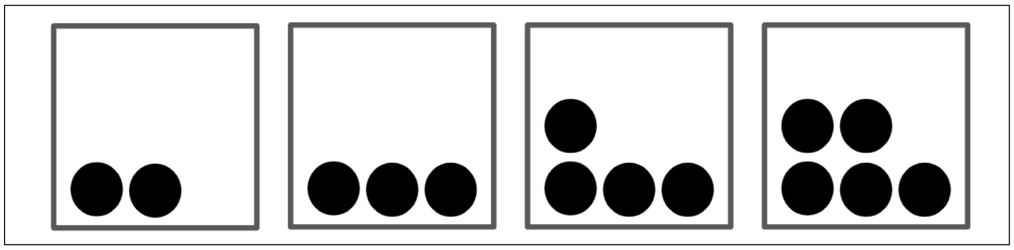


Figure 3.2: Numerical representations of the heap pattern where dots are arranged filling each row from left to right and bottom to top.

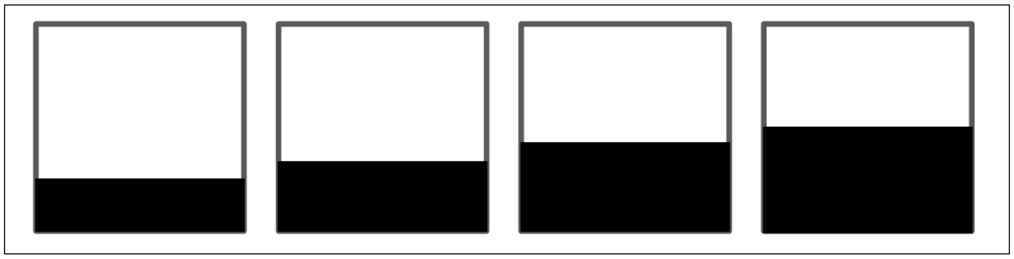


Figure 3.3: Numerical representations using a rectangle that grows from bottom to top.

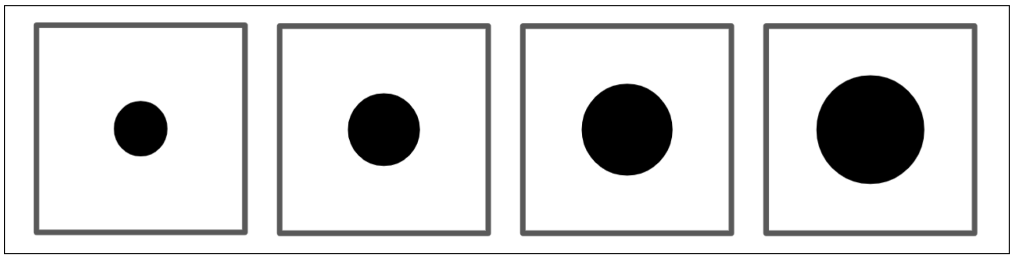


Figure 3.4: Numerical representations using a circle that grows in diameter.

3.2.2. Numerical Order

The studies introduced both adjacent and non-adjacent sequences, but while the research from Inoue et al. [3] and Silberberg et al. [5] used the ascending order, in the study by Barbay et al. [2] the task consisted in choosing the highest value within a set of numerical representations. In order to enable simulating all previous solutions, the software presents the following two options:

- **Increasing:** During a test using increasing order, the correct sequence goes from the lowest to the highest numerical representation.
- **Decreasing:** This numerical order means that the subject has to choose from the highest to the lowest numerical representation.

3.2.3. Extensive log of interaction

During the studies from Inoue and Matsuzawa [3] and Silberberg and Kearns [5], the data was recorded by their (proprietary) software and, while they do not describe in detail the format of the logs, during the experiments the data was locally stored.

In the case of `InCA-WhatIsMore` [2], the logs were stored within the browser's local storage. The information collected about each test consisted of the representation used, the set of numerals to choose from, the value selected, the answer time, and the date, among other parameters. An issue with using local storage only is the added cost of gathering data and an experimenter refusing to send their data.

The software developed in this thesis stores data in the browser's local storage, but also sends it to a remote server, facilitating the researcher's task to access the data from multiple devices. This approach reduces the time it would take to manually retrieve the logs from multiple physical setups and reduces the risk of any "selection bias"where experiments omit to share some data (e.g. because "the subject was in a bad mood").

3.2.4. Synthesis

Table 3.1 summarizes the relation between the three studies described above and those seven features.

3.3. Design

For the application to function in a way that made adding new representations and features easier, some principles of software development were taken into consideration (see Subsection 3.3.2). Thus, functionalities were divided into three modules, which are described in the following subsection.

	Inoue and Matsuzawa [3]	Silberberg and Kearns [5]	Barbay et al. [2]	Software Developed
Local log	✓	✓	✓	✓
Remote log				✓
Increasing order	✓	✓		✓
Decreasing order			✓	✓
Arabic	✓	✓		✓
Dice			✓	✓
Heap			✓	✓
Rectangle			✓	✓
Disc			✓	✓
Matrix resize				✓

Table 3.1: Comparison of the functionalities featured in the three studies considered.

3.3.1. Architecture

The mentioned modules are the Drawer, the Match, and the Game handler, whose functionalities are described as follows:

- **Drawer:** The main functionality of this component is to plot the representations and handle the interactions from the user, also allowing an extensible amount of representations to be plotted since this component also acts as an adapter. After each selection, this component forwards the number selected by the user to the Match Handler.
- **Match Handler:** This component is in charge of generating an array of random positions on the screen where the representations will be plotted and, after each interaction, decide whether the user selected the right one. Then, forwards that decision as a boolean value to the Game component and informs the Drawer that the selected representation should not be plotted anymore.
- **Game:** At last, this component handles global variables such as the score, and number of interactions by the user, the array of numbers to choose from, and so on. It generates and resets those variables before and after each round. The most important method within this component is deciding whether a round is over or if it should continue.

Each component is meant to handle a concrete stage within the flow of information during a round (see Figure 3.5). The mentioned flow embodies the following features needed for the application to replicate the studies that inspired this project (described in Chapter 2).

- A user interacts with the home screen, in which the user can select the representation to play with during the round.
- Before a round starts, the numbers to use during the game should be randomly generated by the Game component.
- That array is sent to the Match Handler, which generates random positions within the screen in which each component will be plotted.

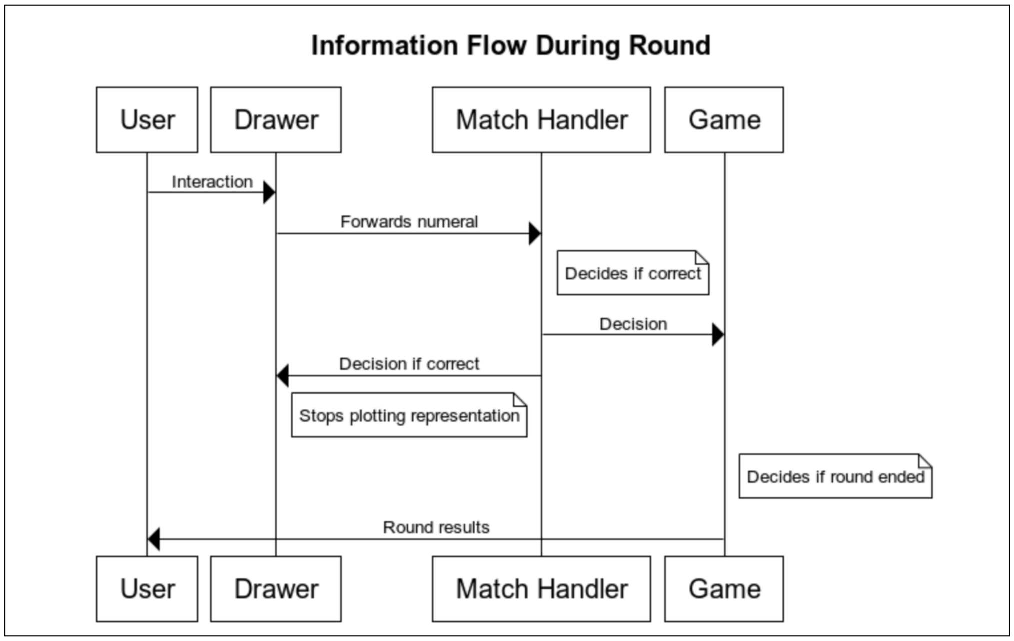


Figure 3.5: Sequence diagram showing the interactions between the user and components during each interaction.

- The Match Handler creates a Drawer for each coordinate generated, and then each Drawer plots its amount using the representation that the user selected on the home screen.
- After each interaction the Drawer component selected forwards its amount to the Match handler.
- The Handler decides whether the choice was right or wrong, in which case it forwards the decision to the Game component and, if necessary, tells the selected Drawer it should no longer be plotted.
- In Either case whether the round should be over or not is up to the Game component to decide. If so, the game variables are regenerated and a new round can begin.

3.3.2. Considerations

To achieve the extensibility required for this project, some principles of software design were taken into consideration throughout the planning and development. Before development started, and during sprints that added big functionalities, the design patterns to use were studied and selected, as a means of turning a complex interaction between classes into 3 interconnected elements similar to the model-view-controller.

The main methods of the application were divided into well-defined components that allowed better maintainability. This made adding new features easier since each behavior within the flow of information was encapsulated and made reusable. The reusability of the code was tested by other students who used the repository from the project to create new `InCA` applications.

As in every software development cycle, bugs appeared along the way, and the module-orientated design eased the fixing process. Writing documentation was also straightforward since each functionality was described within its corresponding component.

The validation of the software developed during this project was part of the weekly sprints workflow, where, before adding any changes to the production environment, the new features had to be validated by Jérémy Barbay. The professor had the role of a product manager, stating new issues to be implemented and validating that the changes were suitable to be deployed.

3.4. Methodology

3.4.1. Weekly sprints

Each week a meeting, with Jérémy Barbay, was held using Discord, where the following topics were discussed:

- The new features added to the application. In order to ensure that every new feature was reviewed, a list of what changed was sent along with the link to the new version.
- The next steps or issues to be developed during the next weekly sprint were set during this meeting. This reunion allowed questions about the reasons and expected behavior of each feature to be implemented, therefore improving the understanding of the project's needs. Each issue or feature had a priority attached, which reflected the level of urgency. Said priorities are:
  - **High:** Also called hot-fixes, this priority is aimed at application-breaking issues like an error resulting in the local log not being filled or the sound feedback not reproducing.
  - **Medium:** Features that are needed to reproduce the studies from Chapter 2. These new features to be added have a medium priority and are meant to be done within three weeks depending on the complexity.
  - **Low:** These features add light quality of life improvements like sorting the settings into self-explanatory sections or adding out-of-the-scope functionalities like a campaign mode. Some of the low-priority next steps are left as future work since they do not impact the MVP but improve the user experience.
- Doubts about the redaction of this document were also addressed during the meetings. When worries related to how to approach the different sections of this redaction appeared, the professor offered his help.

3.4.2. Developed enviroments

In an effort to have a validated version of the application available for the user to use, while also having a development site to test new features; two environments were used during the project. Having separated environments allowed me to have a version of the application where I could add new features and not impact the users. Said environments have their own set of version numbers as explained below, but both are made from 3 numbers separated by a dot (e.g. 1.3.24).

- **Production environment:** This environment contains a version of the application that has already been validated by the professor. Version numbers starting with 1 are used here and with every big change the second number increases, but small changes are reflected in the last digit (e.g. 1.5.2).
- **Development environment:** This other environment contains new features that have not yet been validated, so it could be unstable or contain changes that will never see the light. In this case, the version numbers start with 0 and the other ones work in the same way as already stated (e.g. 0.2.8).

Having this versioning method allows the researcher to quickly report bugs providing information about what changes caused the malfunctioning.

Chapter 4

Implementation

In this chapter, the different features developed for the application are shown and discussed. These are categorized into game functionalities (see Section 4.1), learning-related functionalities (see Section 4.2), and personalization functionalities (see Section 4.3).

4.1. Game

During a normal round an instructor can choose between the three different modes available, each one meant to challenge users in different stages of the learning process. These modes are:

- **Learning mode:** Playing with this setting means that an interaction with a representation only makes the selected one disappear if it is the right choice, as shown in the transition from Figure 4.1 to Figure 4.2.
- **Limited hold mode:** This mode means that all the representations will be covered after the first correct selection, as seen in the transition from Figure 4.4 to Figure 4.3. Selecting an incorrect representation will not cover the figures, giving the user an indefinite amount of time to memorize the current correct sequence.
- **Masking task mode:** When a session starts, and after a fixed amount of time, the representations are covered. This mode is the most difficult out of the ones developed and provides a challenge that can be personalized to suit the capabilities of the user since the amount of time before masking can be configured in the settings.

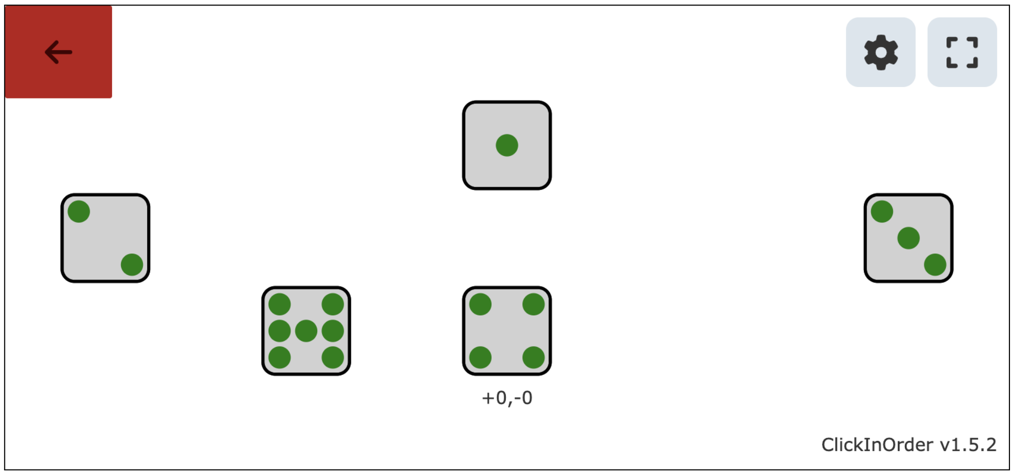


Figure 4.1: Game session using the dice representation and the learning mode.

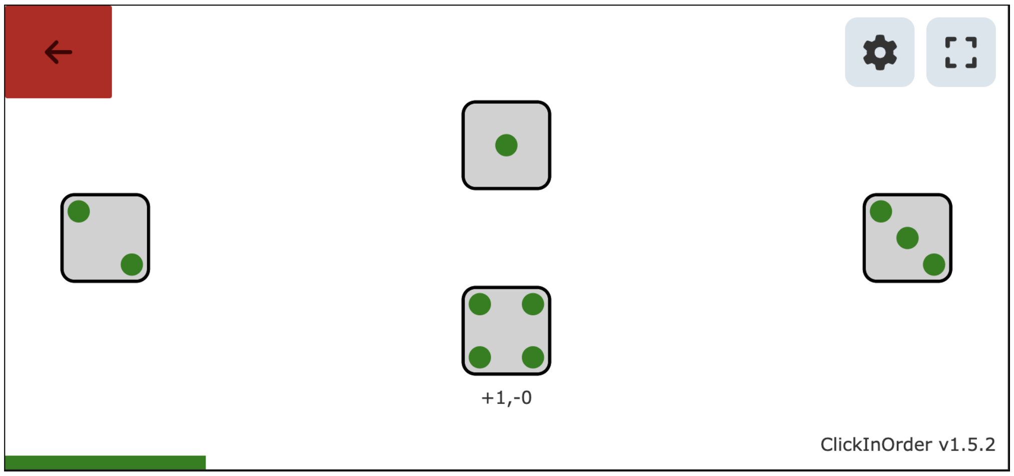


Figure 4.2: State of the game from Figure 4.1 after the first correct interaction.

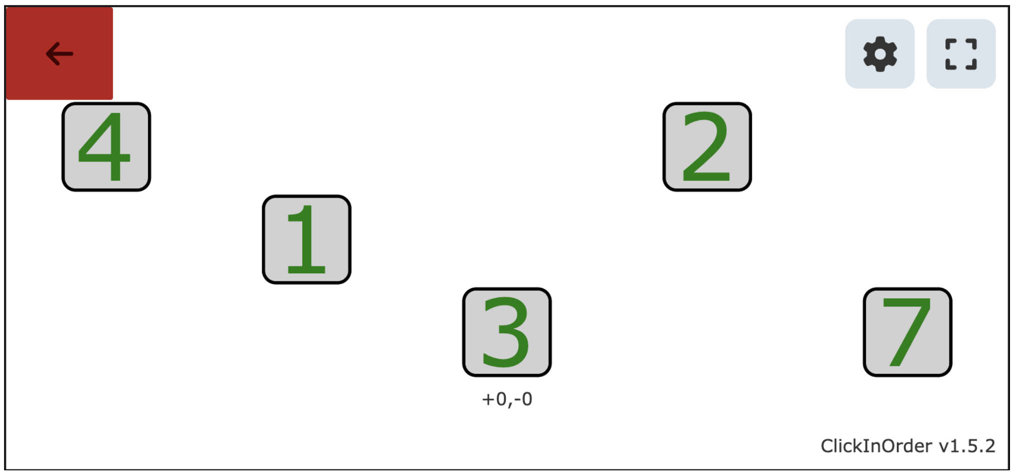


Figure 4.3: Limited hold task session with the Arabic representation before the first interaction.

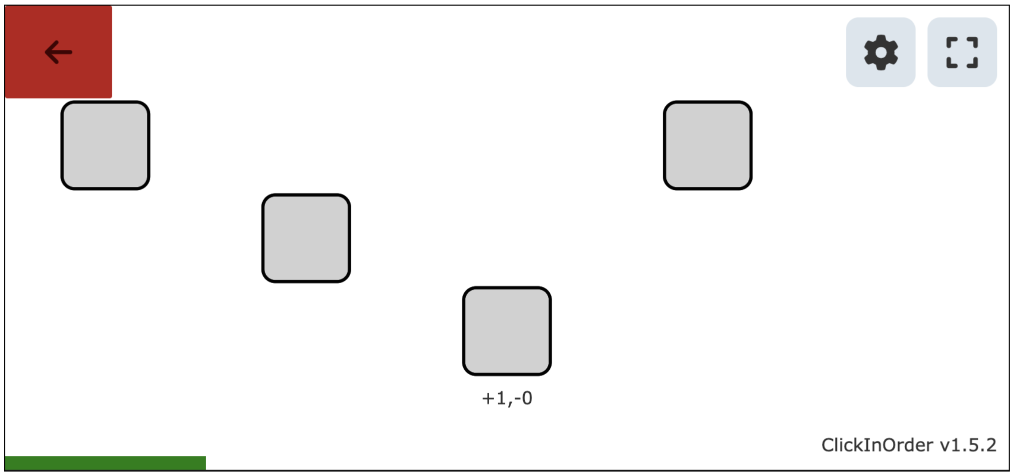


Figure 4.4: The state of the game from Figure 4.3 after the first correct interaction from the subject (in decreasing mode, after 6 was selected).

After each interaction with the application, a logging event is triggered, and, after locally storing that information, a request is sent to Firebase with a JSON containing the details about said interaction. This method aims to reflect what a researcher would do if they were observing a training session. As a means of achieving this goal, each log contains the following information:

- **Timestamp:** Date and time of the occurrence of the interaction.
- **Teacher:** Name of the user's supervisor.
- **Subject:** Name of the user.
- **Action:** Name of the action made by the user.
- **Details:** Aditional information from each action.

This format allows a wide range of interactions to be described, stored, and subsequently analyzed. The next actions are currently supported:

1. **Mode selected:** The details contain the value and type of representation within the menu from left to right. Also adds the selected one and the visual configuration, describing the background and foreground color, and opacity of the representations.
2. **Game started:** An array containing the values and the corresponding order is saved in the details.
3. **Game ended:** The number of correct and incorrect interactions and the accuracy are saved, along with the number of microseconds that the user took to finish the round. At last, an array containing each selection that was made in order of appearance.
4. **Option clicked:** The selected value and a boolean expressing the result are saved.

5. **Game exited:** No other details are added.

6. **Gameboard clicked:** No other details are added.

7. **Menu Background clicked:** This event does not give additional details.

Other features were added to enhance the experience and fit the abilities of each user. Said features are:

- **Threshold:** An error threshold was added, meaning that after a configurable number of consecutive errors, the round will end.
- **InCA-WhatIsMore Mode:** This configuration adds a new layer of personalization within each round. This results in needing a configurable number of correct interactions to beat the round. Therefore, instead of needing to clear all the representations shown on the screen, the user only needs to correctly select n.

4.2. Learning aids

For the sake of providing a suitable learning experience for the user, learning aids were added to the application. These features complement both the visual and audible aspects of the learning process.

The permutation path draws a trail made of triangles guiding the correct order to follow as visible in Figure 4.5. Since the game allows both increasing and decreasing numerical orders, the base of the triangle indicates which value goes first. It can be enabled in all game modes but is only visible while the representations are not covered.

Sound feedback was also added, providing both the user and supervisor with a response from each interaction. This provides the researcher information without being directly involved in the experiment (e.g., watching the screen) since that could bias the results.

The basic phrases are "correct" and "incorrect", which are reproduced after a right or wrong interaction. A "ready" and "exit" sound is reproduced after clicking the interface buttons shown in Figure 4.6. After the end of a round, and taking into account the accuracy held, an "excellent", "pass" or "fail" sound is reproduced. Finally, if the extended feedback is configured, after each interaction's result sound, the phrase "This was X, now click Y" is reproduced, giving instructions to the user of what got selected and what should be the next.

4.3. Personalization

Plenty of customization options are available in the settings that allow shaping the application to the needs of the user. These options are separated into six categories:

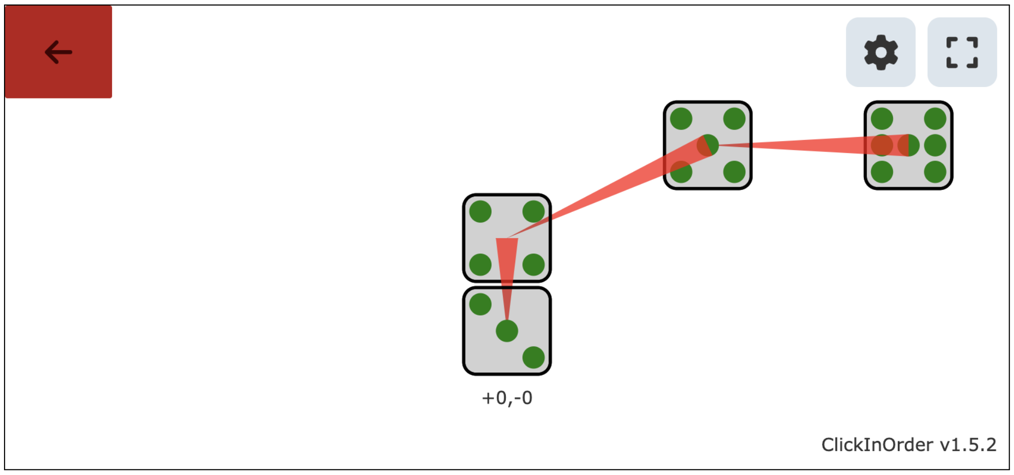


Figure 4.5: Path made out of triangles that shows the correct sequence to follow.

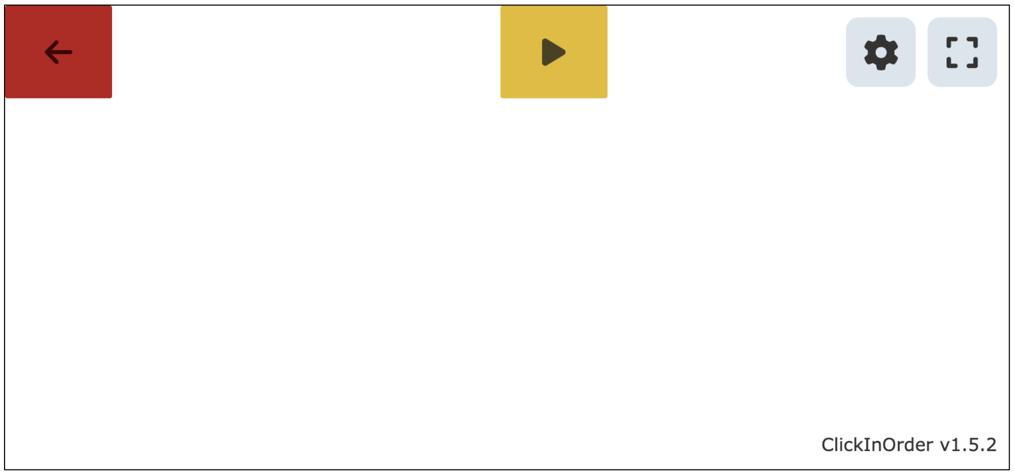


Figure 4.6: Interface with ready and exit buttons that appear before a round starts.

- **Appearances:** The amount and the background color of a representation can be adjusted, along with the background's opacity. As a way of facilitating the supervisor's visualization of how the result will end up being, an example representation is shown as visible in Figure 4.7.

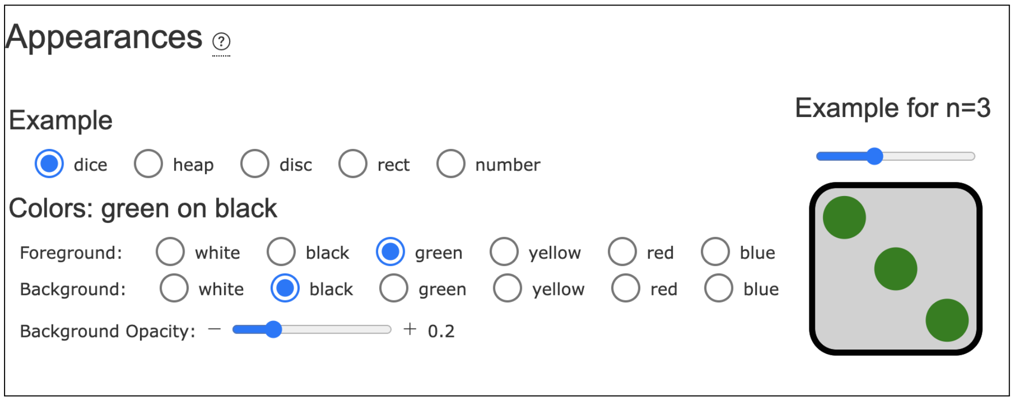
- **Home menu:** Some features were added to the menu, allowing the representations to be randomly ordered, or having a fixed value instead of aleatory ones as shown in Figure 4.8. This was added as a means of checking whether a user is purposely choosing a specific mode or if they choose whatever is in a concrete position like the second mode on screen or just randomly select a mode to play with. The data collected from the menu selections is in itself an interesting topic and could be studied on its own. With the configuration shown in Figure 4.8 the main page will result in the one from Figure 4.9.

- **Game features:** The correct order during a round, and difficulty mode can be tweaked, in the case of the masking task mode, the microseconds before masking can be tuned. The supervisor can also turn on the permutation path (see Section 4.2), the threshold, and the `InCA-WhatIsMore` mode (see Section 4.1). These settings (see Figure 4.10) aim to create jumps in difficulty between each other, making the game experience broadly different rather than small changes.

- **Game personalization:** The set of values and representations to choose from can be selected, along with the number of values to use during a round. These settings enable the supervisor to teach the user little by little, starting with a single representation, a small set of values, and maybe 2 values to choose from within a round and end up using all the ones available, creating small steps for the user to overcome. The last personalization feature available in this category is choosing the number of rows and columns to use during the rounds (see Figure 4.11).

- **Sound features:** The voice pitch and rate can be tuned, and, in some devices that allow it, the supervisor can choose the voice to use. Every single feedback phrase can be personalized and, as with the appearance settings, be tested (see Figure 4.12).

- **Other features:** This last category of settings includes the accuracy needed to have an excellent result or pass the round, the waiting time between each selection, the pressing time needed to use the settings or fullscreen button, and whether to redirect to the main page after finishing a round or not. This group of miscellaneous options gives detailed personalization to the interaction with the interface and the game (see Figure 4.13).


Appearances

Example

[✓ dice] heap disc rect number

Colors: green on black

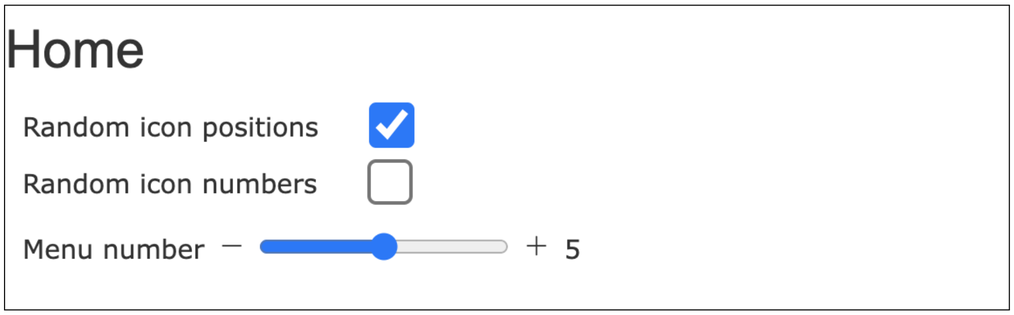
Foreground: white black [✓ green] yellow red blue

Background: white [✓ black] green yellow red blue

Background Opacity: − + 0.2

Example for n=3

Figure 4.7: Settings to customize the appearance of the representations within the home screen and game sessions.


Home

Random icon positions ✓

Random icon numbers ☐

Menu number − + 5

Figure 4.8: Home menu settings where the user can customize the configuration shown on the main page. The result of these settings is the main page from Figure 4.9.

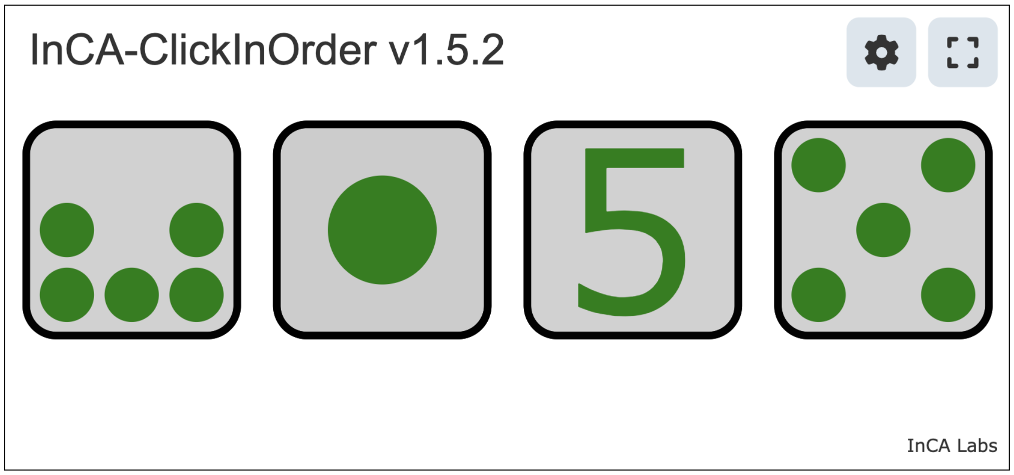
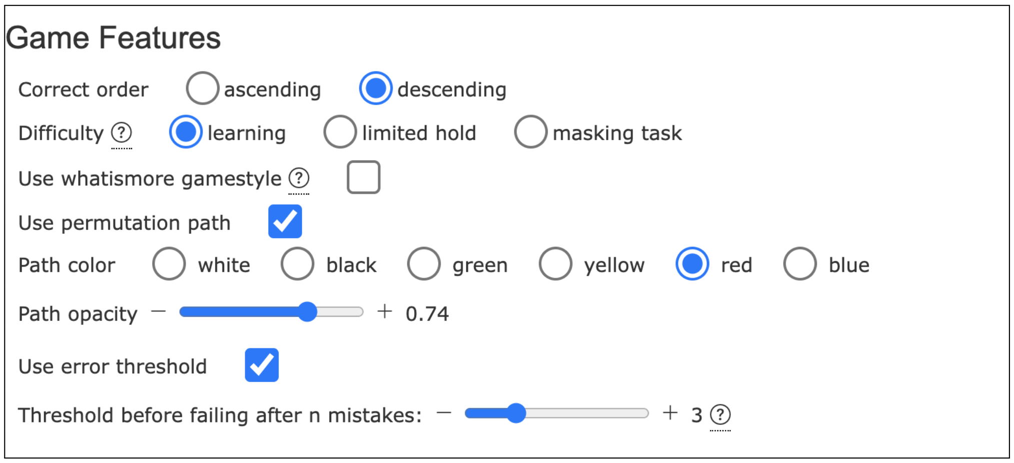


Figure 4.9: Main page of `InCA-ClickInOrder`, with the representations in random positions and only showing the numeral five.


Game Features

Correct order ○ ascending [✓ descending]

Difficulty ? [✓ learning] ○ limited hold ○ masking task

Use whatismore gamestyle ? ☐

Use permutation path ✓

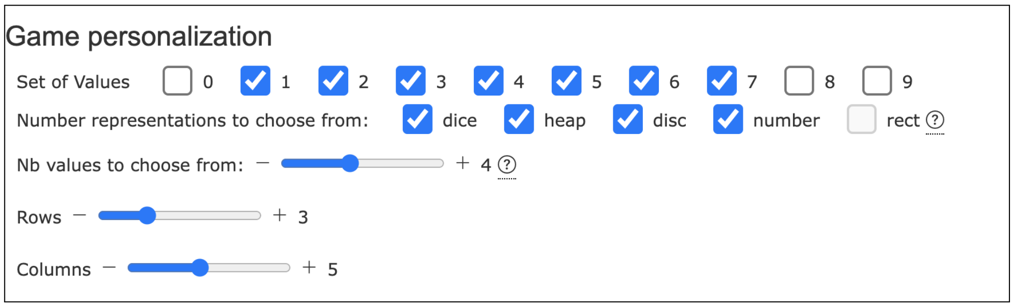
Path color ○ white ○ black ○ green ○ yellow [✓ red] ○ blue

Path opacity − + 0.74

Use error threshold ✓

Threshold before failing after n mistakes: − + 3 ?

Figure 4.10: Settings to adjust the difficulty of the game. Each feature listed is meant to create a leap in the challenge presented to the user.


Game personalization

Set of Values ☐ 0 ✓ 1 ✓ 2 ✓ 3 ✓ 4 ✓ 5 ✓ 6 ✓ 7 ☐ 8 ☐ 9

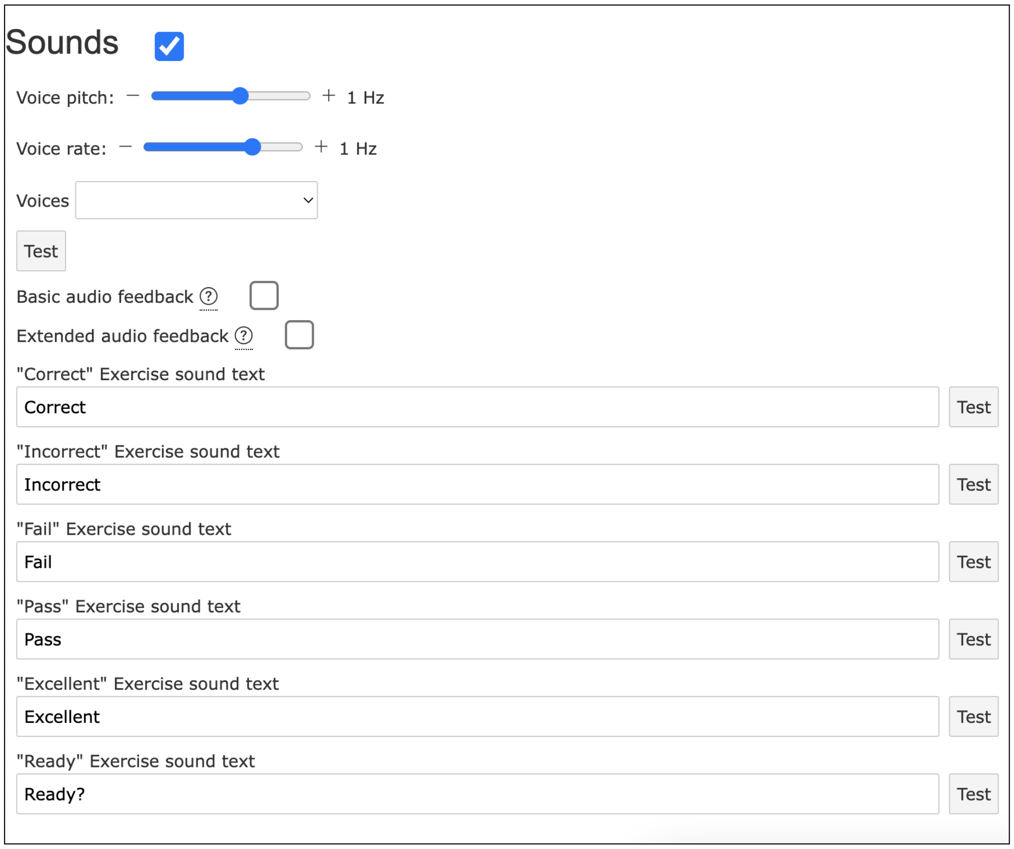
Number representations to choose from: ✓ dice ✓ heap ✓ disc ✓ number ☐ rect ?

Nb values to choose from: − + 4 ?

Rows − + 3

Columns − + 5

Figure 4.11: Game personalization settings that allow tweaking what representations are shown, how many, and the shape of the matrix to use during rounds.


Sounds ✓

Voice pitch: − + 1 Hz

Voice rate: − + 1 Hz

Voices

Test

Basic audio feedback ? ☐

Extended audio feedback ? ☐

"Correct" Exercise sound text

Correct — Test

"Incorrect" Exercise sound text

Incorrect — Test

"Fail" Exercise sound text

Fail — Test

"Pass" Exercise sound text

Pass — Test

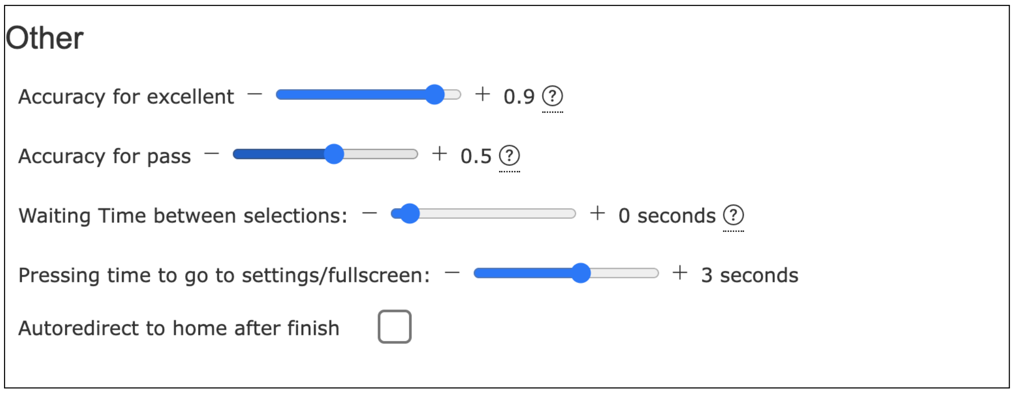
"Excellent" Exercise sound text

Excellent — Test

"Ready" Exercise sound text

Ready? — Test

Figure 4.12: Sound personalization settings with options to customize the audio feedback phrases and test it without starting a new game session.


Other

Accuracy for excellent − + 0.9 ?

Accuracy for pass − + 0.5 ?

Waiting Time between selections: − + 0 seconds ?

Pressing time to go to settings/fullscreen: − + 3 seconds

Autoredirect to home after finish ☐

Figure 4.13: Other personalization features that allow modifying specific interactions with the interface.

Chapter 5

Validation

This chapter contains an analysis of the resulting data gathered with graphs showing how the subjects performed with different representations and numerals. The logs are composed of sessions from July 2023 to January 2024 using the learning difficulty mode. Also, outcomes noticed during the development process will be stated and discussed.

5.1. Results

5.1.1. Mode selection

As already stated, the subjects who tested the application are Lorenzo and Tina, a pair of Quacker parrots previously trained by Jérémy Barbay. The data collected from their sessions includes how many games were played and what mode was used during each session (see Figures 5.1 and 5.2).

It is important to notice that each subject selects the game mode to use during each session. Therefore, one can notice that Lorenzo chose to play with the rectangle mode for more than half of his sessions and Tina selected the disc mode for about a third of hers. This could be due to a personal preference that grew on them, but a specific study could shed some light on why this happens.

5.1.2. Interaction acurracy

Within each mode the amount of correct and incorrect interactions differed, but a main component to take into account when looking at the result is the behaviour of each subject since it appears that, even thought a general behaviour can be noticed, a trend appears when looking at the correct vs incorrect interactions for each numeral.

In Lorenzo's case, the accuracy was better the greater the numeral, as shown in Figures 5.3, 5.4, 5.5 and 5.6. In an overall analysis (see Figure 5.7), one can see that the lower the

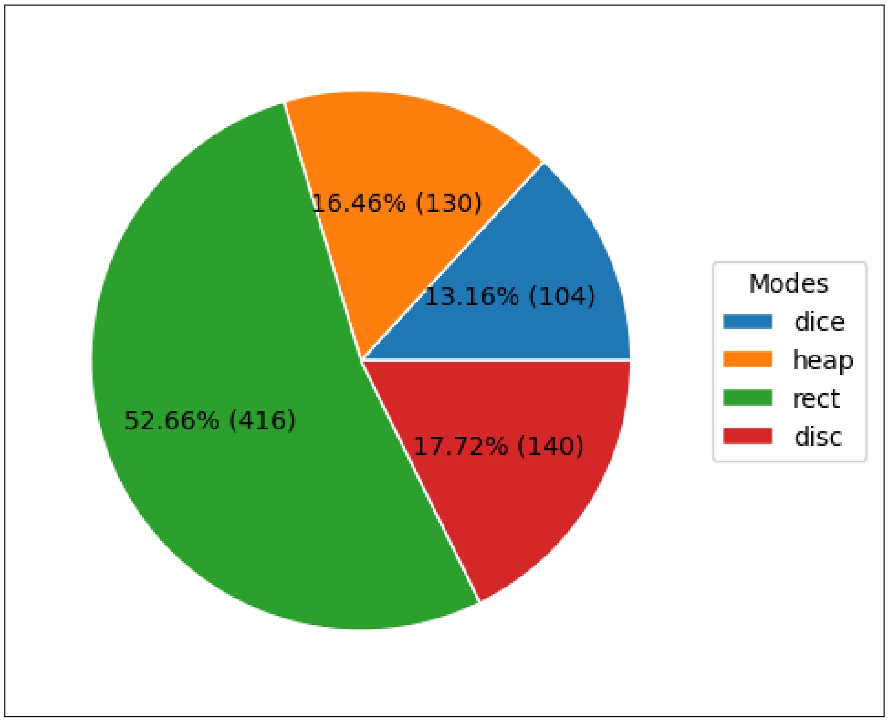


Figure 5.1: Lorenzo's distribution of sessions using each mode where a preference for the rectangle mode is shown.

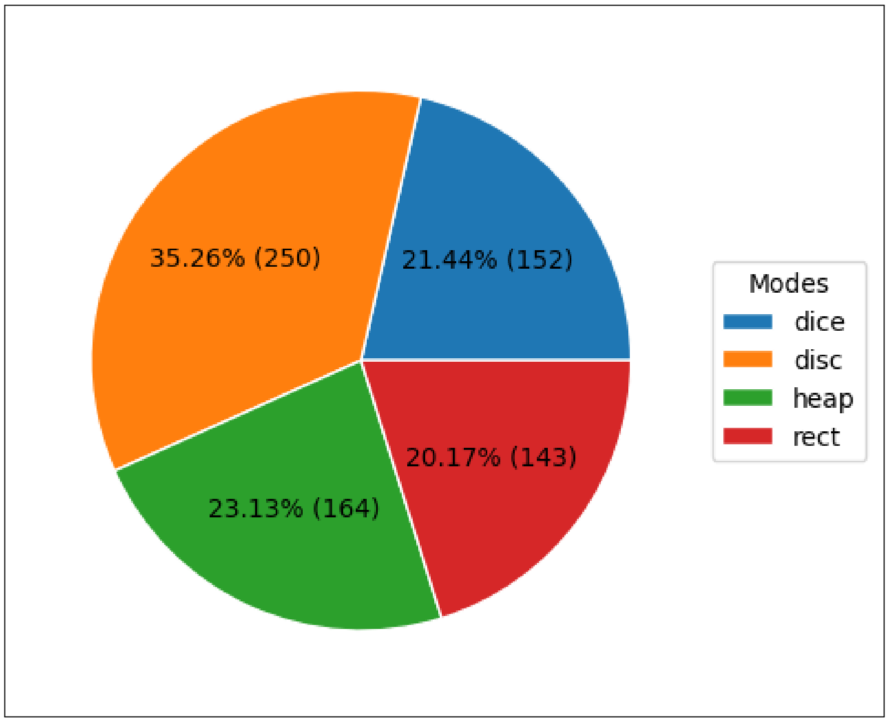


Figure 5.2: Tina's distribution of sessions using each mode where a very slight preference for the disc mode is shown.

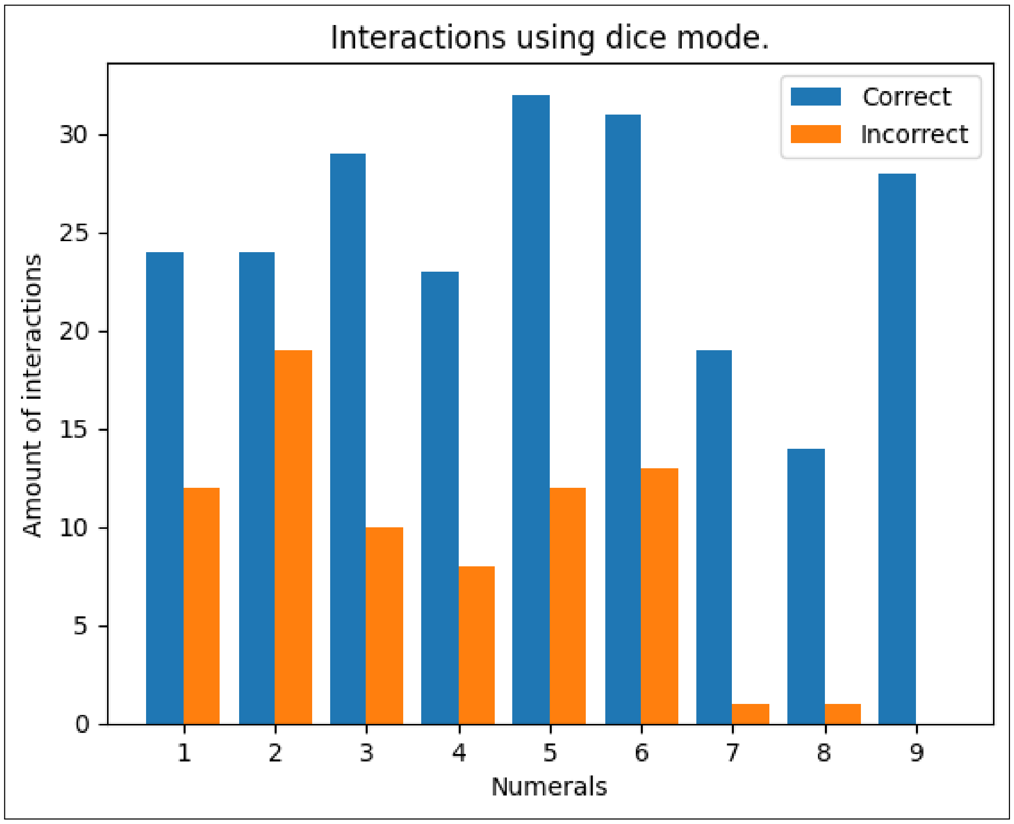


Figure 5.3: Lorenzo's interaction results with each numeral using the dice representation.

numeral, the more incorrect interactions happened resulting in a trend similar to a straight line going downwards. Regarding Tina, she had a similar trend were the highest numerals had the lower rates of correct vs incorrect interactions (see Figures 5.8, 5.9, 5.10, 5.11) but, in this case, the overall behaviour is shaped more like a bell than a straight line as one can see in Figure 5.12.

The results shown translated in an accuracy (correct/incorrect) of 0.822 for Lorenzo and 0.817 for Tina. This means that still after the differences of between each subjects own patterns, their performance was almost the same.

5.2. Outcomes

A lot of knowledge was acquired during this project, not only in the field of coding itself and learning new frameworks but also related to the process of software development made to be used by humans and species other than humans.

- After validation, if a bug was detected, a hotfix had to be made, and only after a new validation, the new features could be uploaded to the production environment. This workflow was difficult to get used to during the first weeks of development since more steps were required compared to when programming a personal or lecture-related

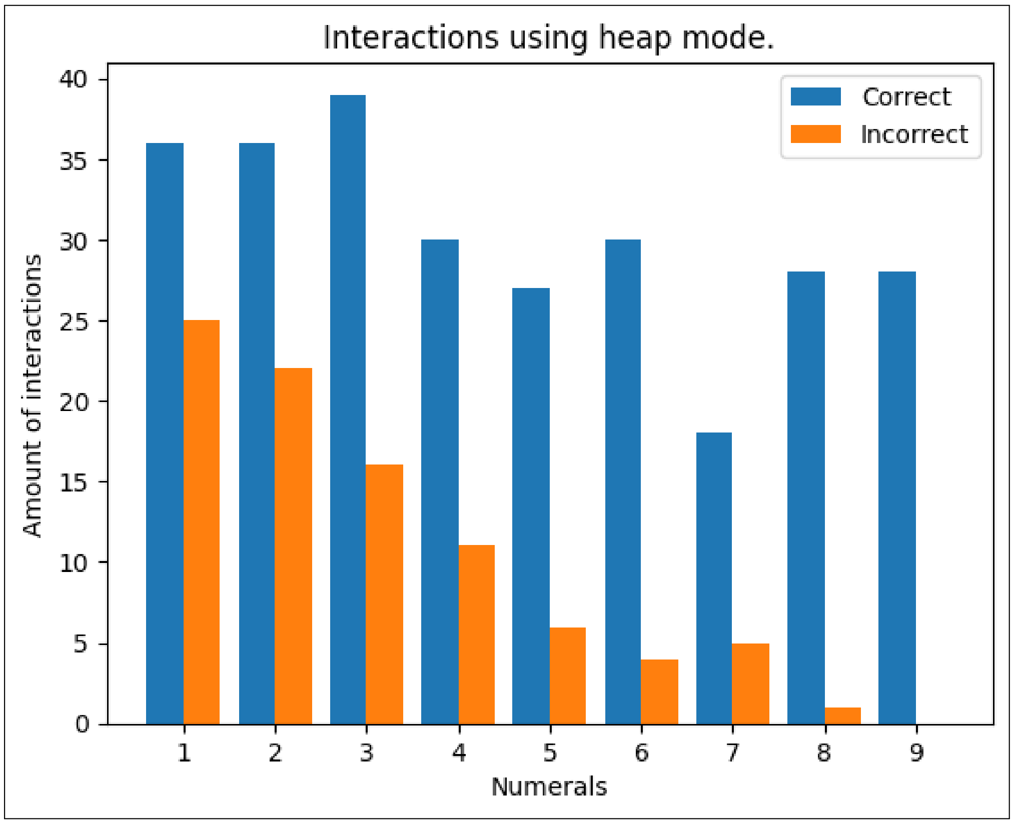


Figure 5.4: Lorenzo's interaction results with each numeral using the heap representation.

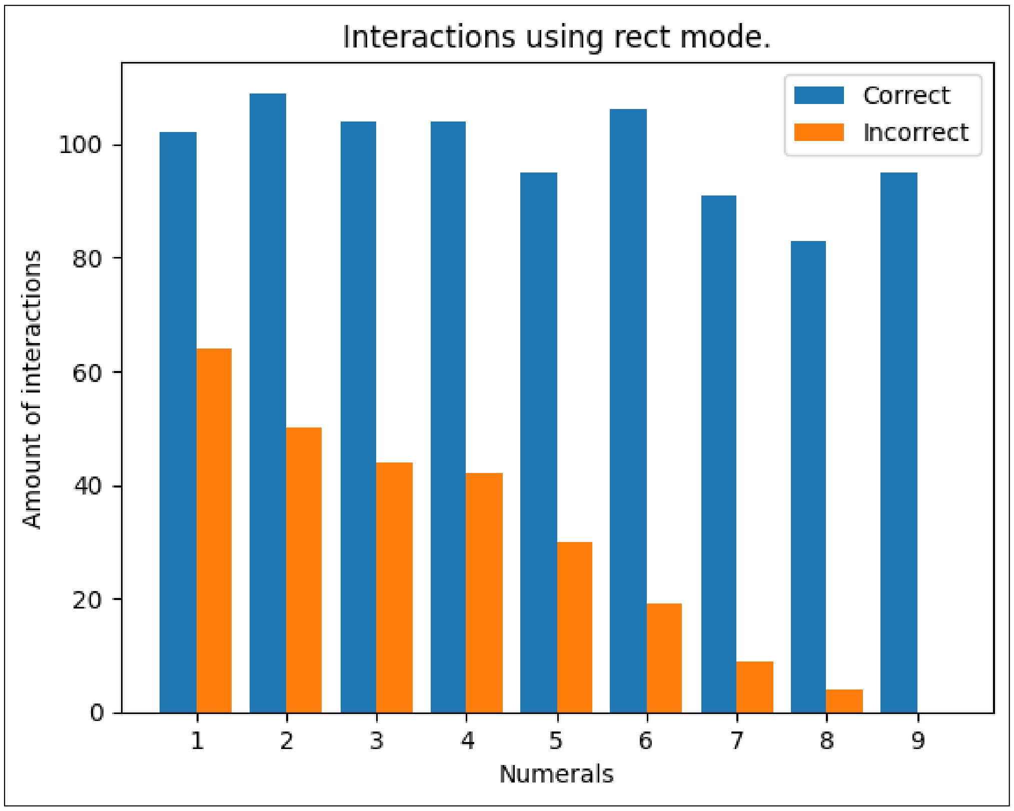


Figure 5.5: Lorenzo's interaction results with each numeral using the rectangle representation.

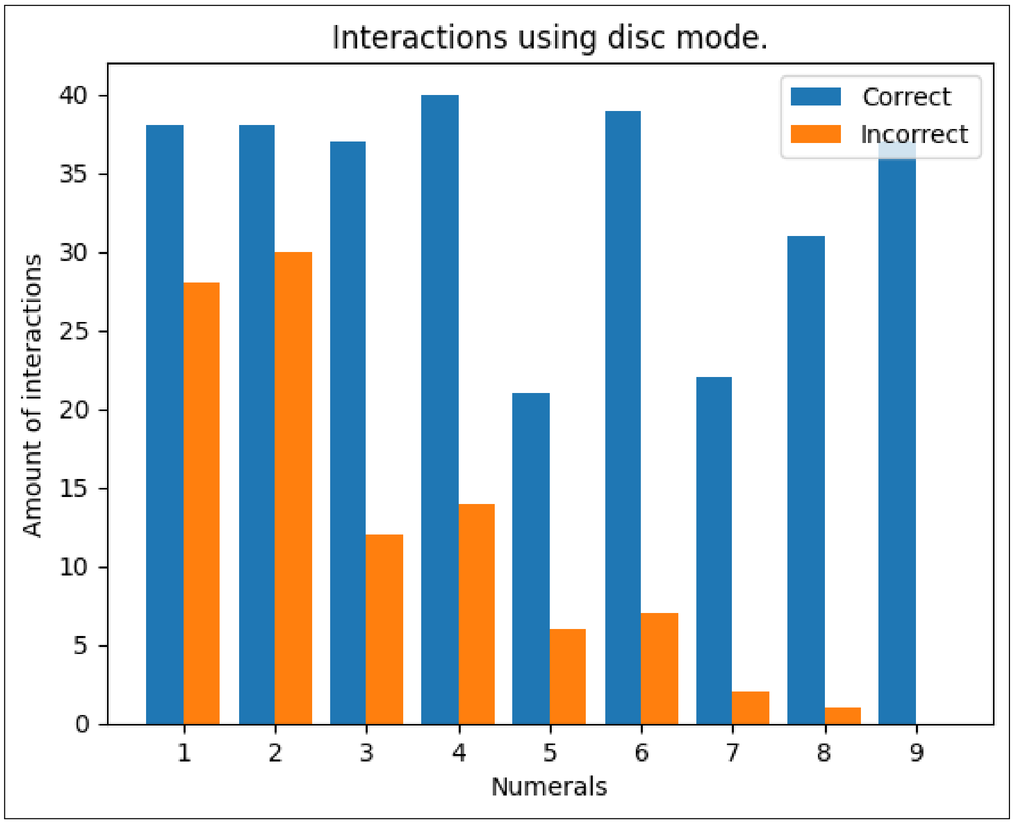


Figure 5.6: Lorenzo's interaction results with each numeral using the disc representation.

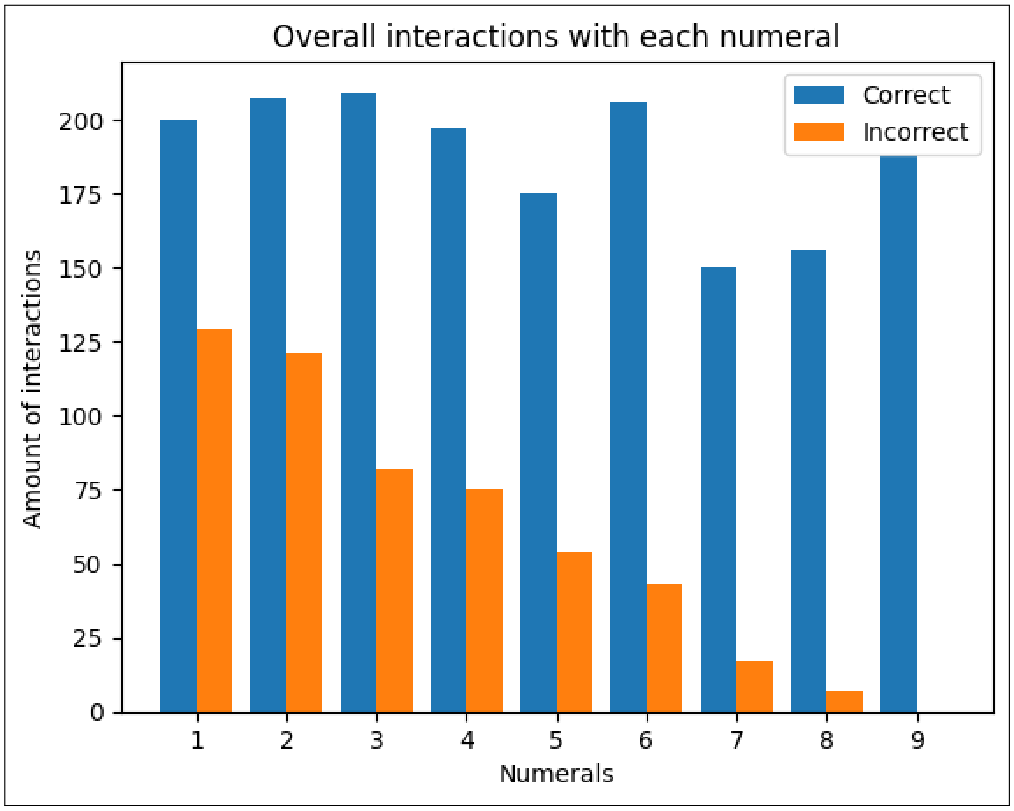


Figure 5.7: Summary of Lorenzo's interaction results with each numeral.

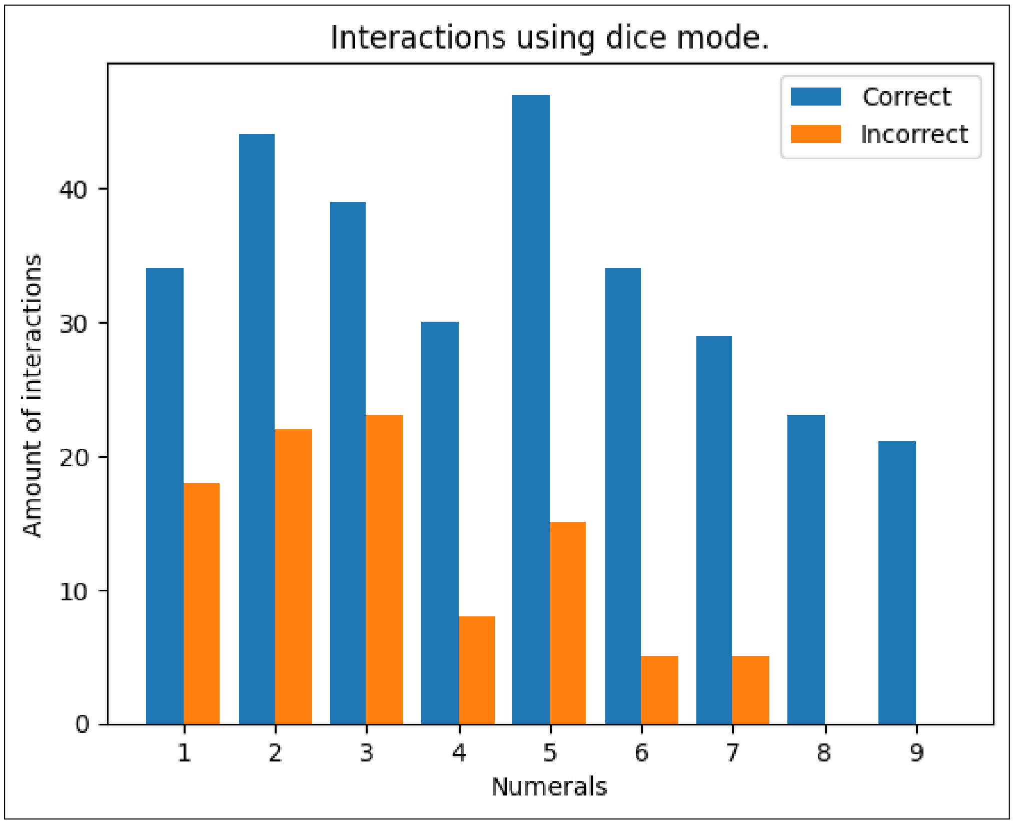


Figure 5.8: Tina’s interaction results with each numeral using the dice representation.

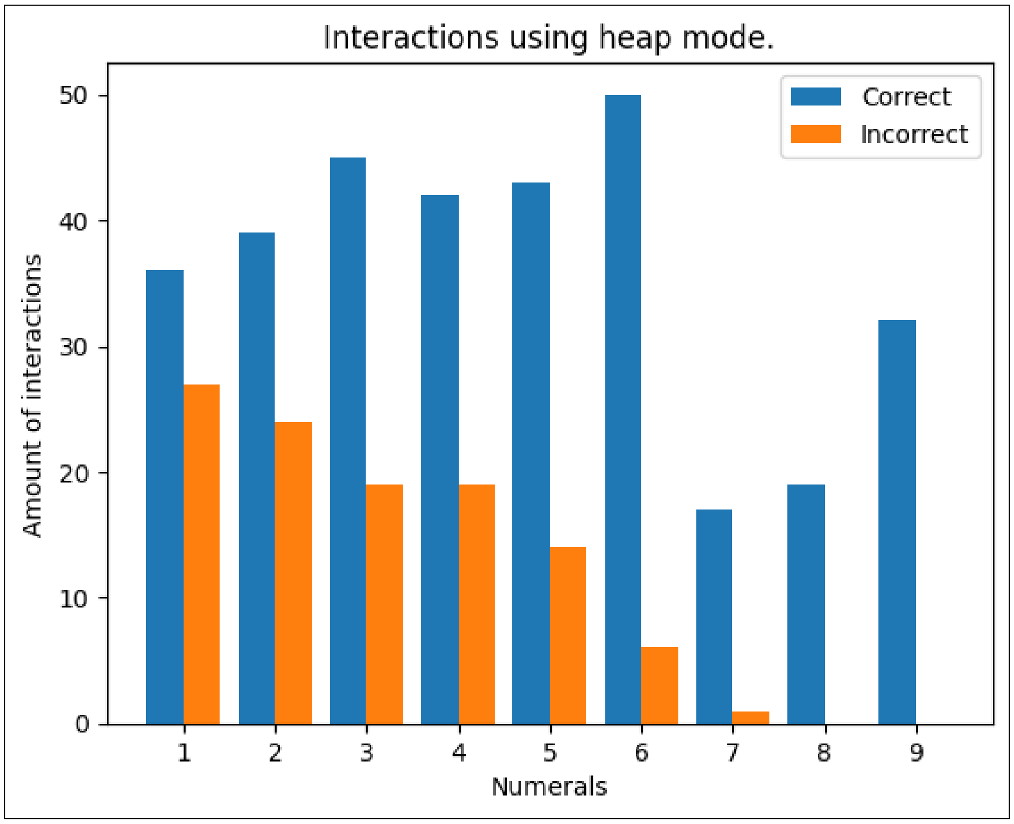


Figure 5.9: Tina’s interaction results with each numeral using the heap representation.

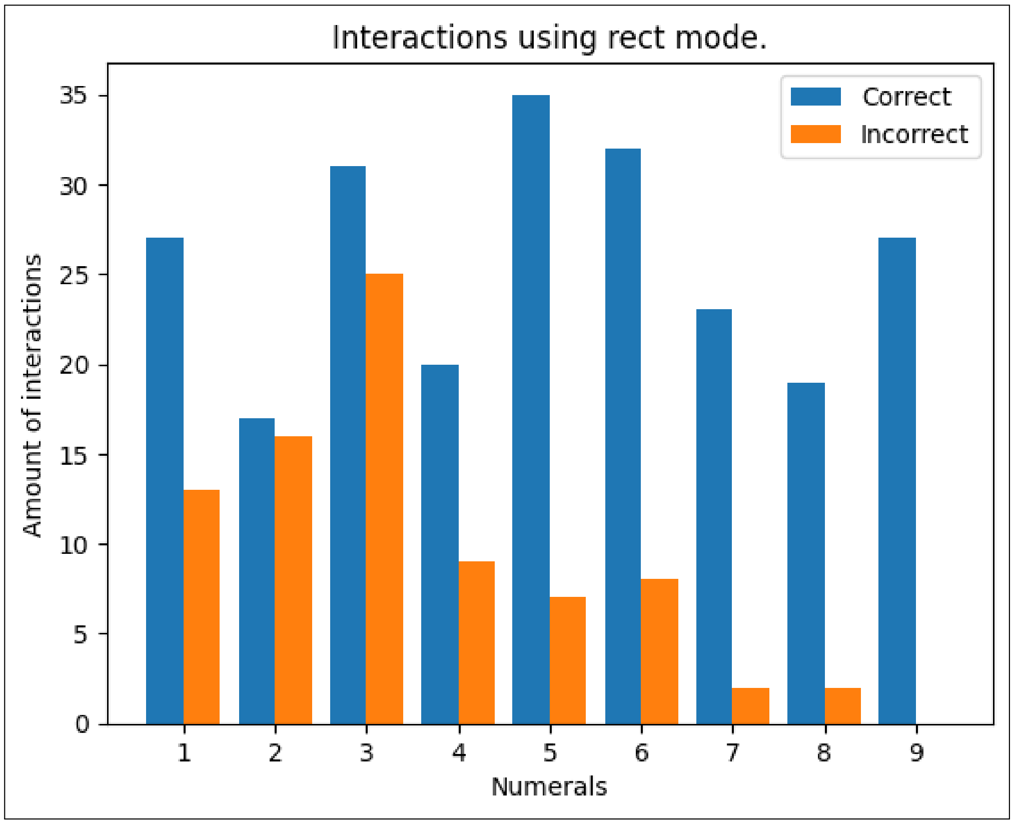


Figure 5.10: Tina's interaction results with each numeral using the rectangle representation.

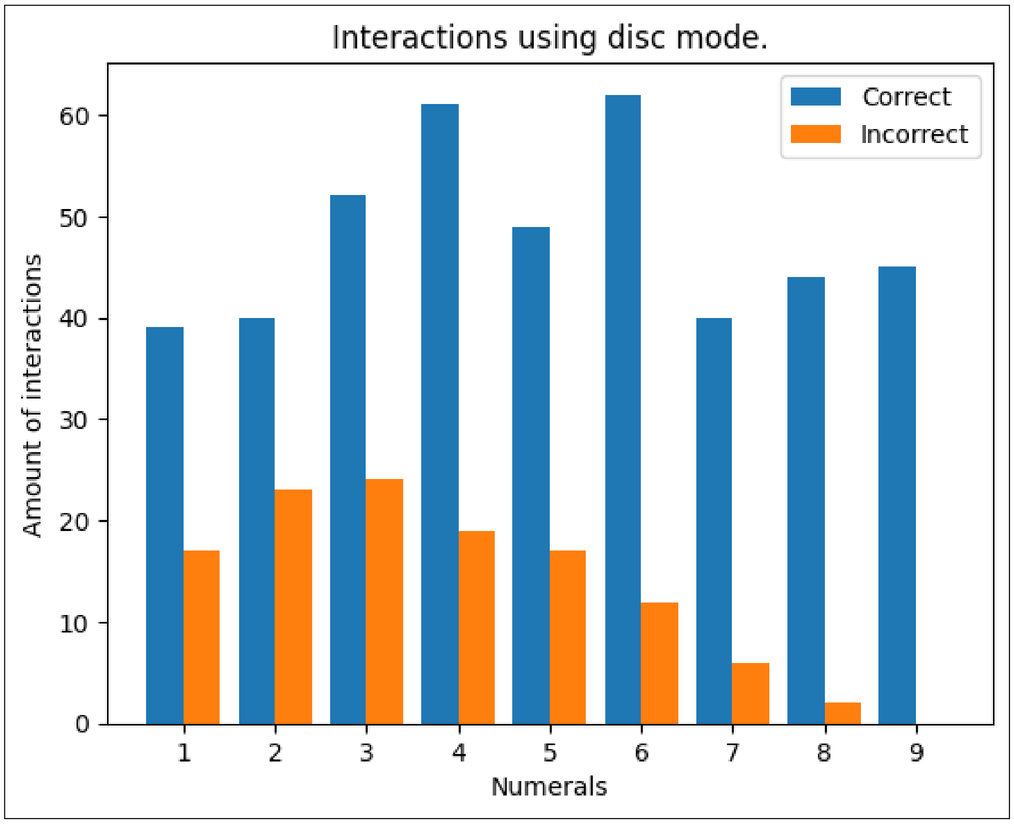


Figure 5.11: Tina's interaction results with each numeral using the disc representation.

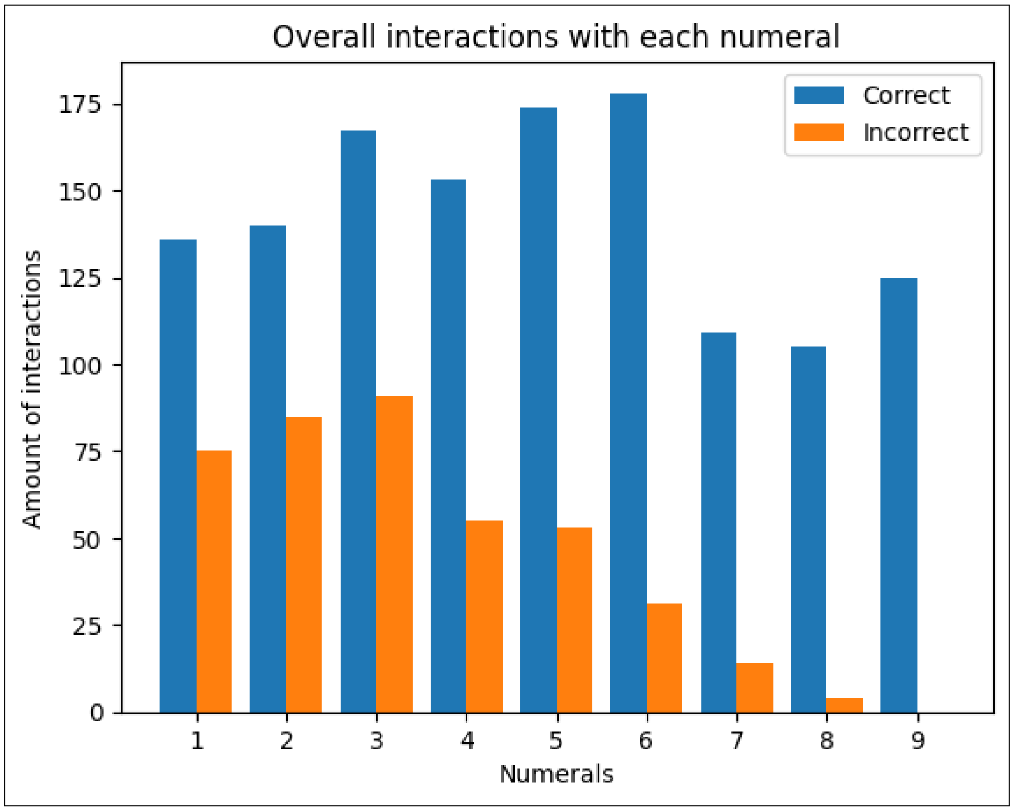


Figure 5.12: Summary of Tina's interaction results with each numeral.

project. But after all, becoming accustomed to more complex workflows is important since it's more aligned with what current software development is like.

- The software created was developed using agile methodologies, thus, some requirements were added or changed during the sprints in order to suit the user's needs. With this in mind, the level of abstraction achieved in the final version of the application was a result of trial and error, perfected after each sprint. The outcome of this specific issue was the learning and application of software design patterns that allowed changing the logic of a component without breaking the functionalities of the others.

- Especially during the last few weeks of the project, the time scheduling got difficult, resulting in less time for writing and delaying sending drafts to the professor. Making a list of the contents of each section before redacting turned the writing into an easier task, also, scheduling a fixed amount of hours per week to work on this document helped a lot.

- Since this project is aimed to be used on mobile devices, components resizing not working as intended made the user experience much worse. In order to solve this issue different approaches were tested:

  – Adding a listener that detected the rotation of the phone and then resized the components, but this failed when the phone got locked and then unlocked. When this happened the browser was not able to receive the rotation signal making the size of the representations off or even unusable.

  - Implementing an event that checks every few seconds the screen's orientation. This achieves the intended behavior and does not consume many resources, allowing the application to resize the representations when required without the need for a tracking variable.

- Some issues related to maintaining the application for multiple browsers appeared during development, each one was hard to track and had to be resolved independently since each browser behaved a bit differently including their mobile and desktop versions. Having issues that were particular to a browser made me aware of having to test in multiple configurations before releasing an update, and a few of them were:

  - The option to customize the voice from the audio feedback, since every browser has an array of voices to select from. This feature raised a problem when the list of voices was not accessible from mobile devices, and, after researching, it turned out to be that each smartphone only allows a unique voice that is configured in the device settings. In the end, that feature was kept even though it is only usable on desktops.
  - A concurrent issue was fixing a problem with the fullscreen button, which required the supervisor to click twice before the screen actually went to fullscreen mode. After abundant research on the topic was made, the conclusion was that it was an intended behavior for a browser application. This is due to a mandatory first interaction from the users with the browser before the various events, including the fullscreen event, can be triggered to stop malicious sites. Therefore, to reduce the impact on the user's experience, adding a title screen was left as future work.
  - An audio feedback bug that was particularly difficult to track down was one that made the voices have a noticeable delay between the interaction and the sound. This particular one turned out to be a case of an architecture problem, due to the sound behavior being encapsulated within a class, this singleton was passed to the components that required sound feedback. Having this singleton made the program have a delay when calling the *speak()* method. Therefore, a refactor was required that turned the class into a variety of imported methods that worked faster and thus, had no delay when using the sound feedback

- Many issues appeared whenever a new feature had to be tuned to avoid frustrating the users. Having animals other than humans as subjects meant that a lot of assumptions when developing the application were not actually the best approach. Some of the challenges that were overcome are:

  - In the case of birds, whenever Lorenzo or Tina played with the application, they touched the screen in a fast and consecutive manner, meaning that if an interaction was incorrect, the outcome would not be fair if each interaction was taken into consideration. Therefore, tunning what the applications count as an incorrect selection was a bit of trial and error and thus, the resulting logic was not counting consecutive incorrect selections of the same numeral.

    An important argument that resulted in this behavior was seeing how at times Lorenzo and Tina selected the correct sequence of numbers really fast, and thus, having a timeframe where the users could not interact resulted in frustrating the users. For that reason, a logic that solved having fast incorrect interactions by adding an invincibility timeframe did not suit the needs of the users.

The resulting approach could be seen as too permissive but it was shaped that way in order to fit the needs of the users and after really considering the way the users interacted with the interface.

– The application's reaction time was something that had to be meticulously tunned since a delay in the sound feedback or a clunky change of scenario from a session to the menu could make the users less prone to play with `InCA-ClickInOrder` during their playtime. Hence, having in mind that birds can get more stressed than humans when facing a sluggish interface was something that I learned during the process, and allowed me to experience creating interfaces for users with needs that differed from what I was used to.

Chapter 6

Conclusion

At the end of this project, the data gathered allowed me to deeply understand how the subjects were using the application and, accordingly, validate the objectives of this assignment.

6.1. Summary

The objective of developing an application that allowed reproducing the study from Inoue and Matsuzawa [3] was achieved, the resulting minimum viable product was thoroughly validated resulting in a piece of software that contains the features needed to accomplish the objectives shown in Section 1.3.

As expressed in Chapter 5, Tina and Lorenzo's sessions used four different kinds of representations, the learning mode and up to four numerals, meaning that the more complex experiments where the representations were masked are yet to be carried out. This will require more time for the users to work on their working memory for 2-dimensional permutations. Thus, more investigation can be done, and is currently ongoing, using `InCA-ClickInOrder`, to do further research on the capabilities of Quacker parrots (see Figures 6.1 and 6.2).

6.2. Contribution

All the features introduced in this document were implemented, resulting in a robust application that has plenty of flexibility in terms of adding new content and personalizing what is currently available. With this in mind, the current code has been used by other developers in the `InCA` group.

Not only in the field of research itself, but allowing new people to create applications aimed for animals other than humans was an important outcome of this project. The code written can be extended and used, assuring that all the basic requirements are met.

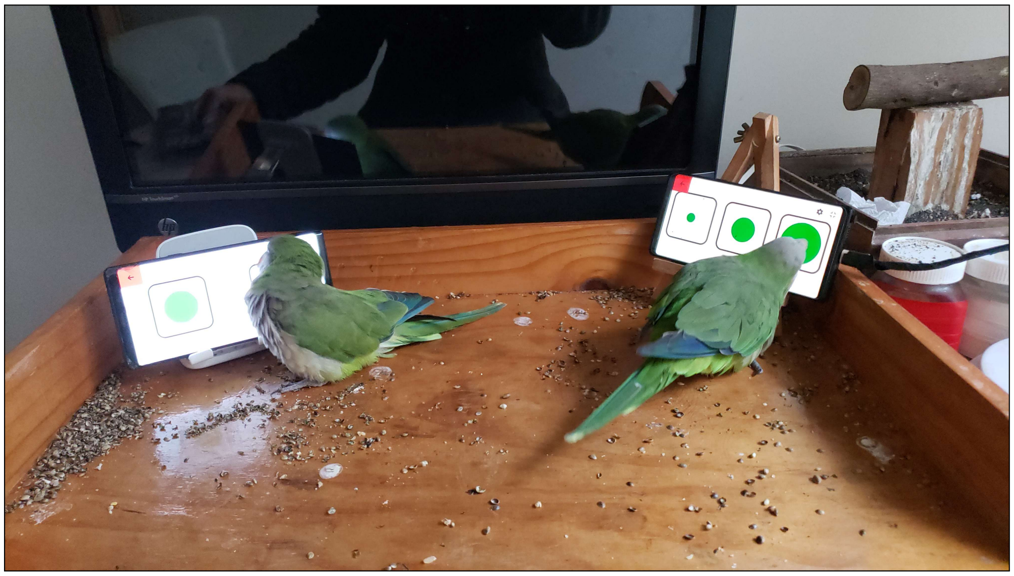


Figure 6.1: Lorenzo and Tina playing with `InCA-ClickInOrder`.

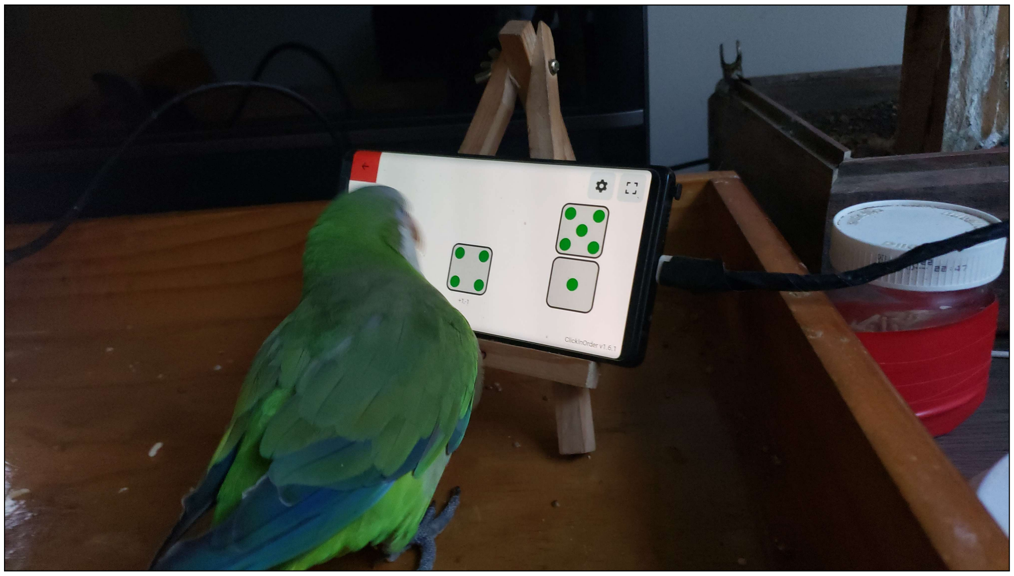


Figure 6.2: Playing session using the dice representation and a matrix of 2 rows and 3 columns.
.

Chapter 7

Future work

With the current state of the application, which contains each feature discussed in this document, the objectives presented in Section 1.3 were achieved. However, the current piece of software in the production environment could be improved, and some suggestions will be explored in this chapter.

7.1. Backlog

Each feature added to the backlog has a suggested implementation path and a possible duration. The limited time given to this project meant that the minimum viable product focused on the mandatory features to reproduce the studies, while some other ideas were not viable within the schedule. Nevertheless, the following extensions would add value to `InCA-ClickInOrder`:

1. **Adding a campaign mode:** Using a spaced repetition approach, a campaign mode could significantly smooth the learning process of a new user. This feature was suggested by the professor, but spaced repetition is my suggested methodology since it is frequently used when learning a new language and, in my experience, is quite helpful. This method, in the context of acquiring a new language, means that if a word's meaning is correctly guessed, it will appear less frequently in daily sessions, but if incorrectly guessed, it will appear sooner. In this particular application, starting with a small range of numerals, after a good performance on several rounds a new numeral will be added to the set, thus, when underperforming the set of numerals that can appear during a session will shrink.

   On top of this, adding a similar progress feature to the number of numerals on each round will result in a robust learning mode that grants the users a unique learning curve that fits their capabilities. Implementing this functionality should take two weeks of exclusive development and another three weeks focused on testing and tuning the mode to ensure the difficulty pacing is in order, assuming full-time availability.

   The data gathered from this mode should be of great interest since the gathered data could be part of a study or thesis related to different learning methods and their impact on animals other than humans. Making said research is something that could extend from six months to a year due to the amount of time needed to gather data that could lead to any conclusion.

2. **Analyzing the logs:** A lot of data was collected during the span of this project and it is stored in Firebase. The mentioned data could be used to replicate the analysis from the studies in Section 2 or to make further investigation on the ability of parrots to discriminate quantities. To do research with `InCA-ClickInOrder` and end up with relevant conclusions about the performance of Quacker parrots or subjects from other species, a researcher will have to gather more data than the currently available, to reflect the learning process of the subjects and check how far they can go in terms of difficulty. This stage of additional training could take around six months, depending on the willingness of the subject to play with the application which, by all means, must be respected.

3. **Improving learning aids:** A feature that could be improved on is the permutation path, in its current state every arrow is displayed at the same time which can be confusing and worsen the user experience. A possible solution is adding an animation to each arrow and displaying only one at a time. Since this issue could mean rethinking the logic behind the implementation of the permutation path, the implementation should take 2 weeks of development, and further testing to see if the users get accustomed or it end up being to distracting.

4. **Improving the user interface:** As previously mentioned, the issue with the fullscreen button means that the supervisor must interact with the interface before being able to go to fullscreen mode. An improvement to the user interface that could positively impact the UX is adding a title screen before entering the home page, this adds a mandatory first interaction without frustrating the supervisor. On the same page, an exploration stage where the objective is seeking UI improvements could potentially detect issues in the users' intended behavior versus their actual course of action when interacting with the application. Since there is uncertainty about how and where to seek UI improvements and a mandatory testing stage to check if the UX improved or not, this stage could take more than 6 weeks and even start a study about the impact of user interface design decisions on the user's performance.

5. **Creating a deployment pipeline:** Automating the deployment is a quality-of-life improvement for the developers but, at last, reduces human errors that could impact what version is uploaded to what environment or the error reporting capabilities of the supervisors if the version numbers are faulted. Since adding DevOps practices to an already established project can produce complications, 2 weeks is the suggested time to complete this feature.

Bibliography

[1] Syrina Al Aïn, Nicolas Giret, Marion Grand, Michel Kreutzer, and Dalila Bovet. The discrimination of discrete and continuous amounts in african grey parrots (psittacus erithacus). *Animal cognition*, 12:145–54, 09 2008.

[2] Jérémy Barbay, Fabián Jaña-Ubal, and Cristóbal Sepulveda-Álvarez. Measuring discrimination abilities of monk parakeets between discreet and continuous quantities through a digital life enrichment application. pages 1–14, Association for Computing Machinery 1601 Broadway, 10th Floor New York, NY, 2022. ACM Digital Library.

[3] Sana Inoue and Tetsuro Matsuzawa. Working memory of numerals in chimpanzees. *Current Biology*, 17(23):R1004–R1005, 2007.

[4] Sana Inoue and Tetsuro Matsuzawa. Acquisition and memory of sequence order in young and adult chimpanzees (pan troglodytes). *Animal cognition*, 12 Suppl 1:S59–69, 09 2009.

[5] Alan Silberberg and David Kearns. Memory for the order of briefly presented numerals in humans as a function of practice. *Animal cognition*, 12:405–7, 03 2009.

ANNEXES

Annex A

Documentation

A.1. InCA-ClickInOrder

This project is part of the **InCA** applications meant for Animal-Computer Interaction research. The template used for this project belongs to Cristobal Sepulveda, who also helped with the connection with Firebase. The objective of this application is to replicate the studies done by Inoue and Matsuzawa with chimpanzees but with Quacker parrots.

A.1.1. Get started

After pulling the contents of this repository you will need to install the dependencies, which can be done using the following command:

```
npm install
```

Then you can run the code and see the application in your localhost[1] by executing this command:

```
npm run dev
```

You might notice that a version number appears in the application content, this can be configured by changing the `version` variable within the `package.json` file. After doing so, you might also notice that the interaction logs are only being saved in the local log, but not sent to Firebase. To do so you will need to change the parameters related to the API in the `rollup.config.js` file.

A.1.2. Deployment

In order to have a ready-to-deploy application you will need to execute the next command:

[1]`http://localhost:8080/`

```
npm run build
```

This will generate the needed files in the folder `public/build`, you might notice that each file generated has a version number appended, this is the one configured in the `package.json` file. This allows not having the new features reflected in your device since the new code might not be added because browsers tend to use cached files to optimize resources.

After this, you might need to modify the `public/index.html` file to use new `.js` and `.css` scripts. Finally, the content of the public folder can be copied into your server and, thus, be available in a production environment that users can have access to.