UTRECHT UNIVERSITY
Graduate School of Natural Sciences

Game and Media Technology master thesis

Adaptive Futuro Cube: Using Dynamic Difficulty Adjustment In Minigames To Train Fine Motor Skills.

First examiner:
Remco Veltkamp

Second examiner:
Julian Frommel

Candidate:
Maik Vink 6797547

September 22, 2024

Abstract

Children suffer more and more from delayed fine motor skills and the use of smart toys could be a solution to reduce this problem by keeping the children physically active. The Futuro Cube is an example of one of these smart toys, which has been successfully used in research to detect delayed fine motor skills in children. The Futuro Cube could serve as a versatile tool when it can be used as both a toy to detect and train these fine motor skills. In this paper, an extensive analysis is performed to see the training capabilities of the Futuro Cube. Using a Dynamic Difficulty Adjustment system, a game can be played that is both fun and adjusts itself correctly so that the user's skills. The Dynamic Difficulty Adjustment system adapts itself so that the user pushes their fine motor skills to new limits. Preliminary results show that the Futuro Cube could be a useful tool for training fine motor skills. However, it will still require some fine-tuning on both the soft- and hardware side. For the software side, the DDA system can be further optimized and the played game on the Futuro Cube can also be further improved. For the hardware side, it will require a more precise detection system. It is thus unlikely that the Futuro Cube will be used in therapy sessions with children anytime soon, but with some additional research, it is possible to become a great training smart toy. Additionally, the Futuro Cube has also shown possibilities that it could be useful for other circumstances, for example, to reduce RSI risks in adults.

Contents

1. Introduction

The Futuro Cube (FC) is a 6-sided cube with a 3x3 RGB LED matrix on each side. The cube is created by Rubiks. The FC can detect motion, gravity, tapping, and even certain gestures. A reference picture can be found in Figure 1.1. Using the program Pawn, we can create and adapt the games that can be played on the FC. Additionally, we can gather and save play data, to analyze it later. In section 2.2 we go into more detail about the hard- and software of the FC.
Its original purpose was to play games on it—e.g. Tetris and Snake[1]. In Snake, you appear as a series of dots that slowly grow as you eat more apples (which is a flickering red dot somewhere on the Futuro Cube). The snake will always move towards the top of the Futuro cube, so by moving it around and rotating it you are able to move the snake around the cube. The game ends when the snake ends up hitting itself with his movement. In Tetris, or the renamed version Cubris, Tetris-pieces are created on one of the sides of the FC (always the same side after the game is started). You can press the side of the cube to turn a piece clockwise or counterclockwise and you can tap the top to place a piece. The FC can detect which face (i.e. side or top) using the gravity cursor built on top of the FC. If you fill a face or you manage to create a ring around the whole FC, the dots clear up and you get points.
These games and the other games are further explained and explored in Appendix A.

The Futuro Cube has also been successfully adapted and used for research purposes. [2]. During one such research, it has been shown that the FC can successfully detect delayed fine motor skills. In this thesis, we want to expand the FC usage by making it also a training device for fine motor skills.

Approximately 5-10% of elementary children have problems with fine motor skills. [3] [4] Part of this percentage comes from stroke survivors and children suffering from cerebral palsy. That being said, there is a growing percentage of children with fine motor skills problems or the danger of having those problems, which is more likely caused by the inactivity of children. Dylan et al. [5] showed that being physically active and training your fine motor skills are closely related.

Having poor fine motor skills is problematic because fine motor skills are detrimental to the child's social skills and learning ability. [6] [7] For this reason, these issues must be detected and treated as soon as possible.

The Movement Assessment Battery for Children Second Edition (MABC-2) [8] is a well-known worldwide used test to detect fine motor skills, which most pedagogues have used. While this test can detect fine motor skills accurately, it does lack an aspect of enjoyment for the children. The Futuro Cube is a possible solution to this problem.

Figure 1.1: Futuro Cube

It has been determined that the Futuro Cube can detect fine motor skills as accurately as the MABC-2 test[2]. With this in mind, we can also take it a step further. It is common for games to be used for rehabilitation purposes, because of the entertaining value behind them. So if we expand the possibilities with the Futuro Cube to not just detect, but also to train the fine motor skills of children, it could reduce the amount of children suffering from fine motor skills problems.

Two important factors should be considered if we want to use the FC for rehabilitation purposes:

- The minigame should keep the user engaged. If the game has low engagement, then the users will be more likely to quit, which is undesirable.
- The minigames should align with the user's current skill level. We cannot expect someone struggling to suddenly become better at a game if they don't get a chance to train their skills. Furthermore, if it's too hard then a child might get frustrated and do something more fun. At the same time, if it is too easy it could either result in the child being bored and stop playing with the FC or not getting challenged enough and end up not training their fine motor skills. All described cases have undesirable outcomes.

Dynamic Difficulty Adjustment could be used to ensure both factors are optimized in the minigames when the users are playing with the FC.

Dynamic Difficulty Adjustment, or DDA, is a form of adaptive gaming. It's a mechanic that is used in games to make sure that the game is not too difficult or too easy. DDA makes it so that a game will become harder if it notices the player finds it too easy and becomes easier if it notices that the player is struggling. This way the game will try to find the perfect difficulty for every player, intending to keep the player engaged with the game.

We want to use this system in the Futuro Cube to keep the users engaged with the Futuro Cube. The Futuro Cube has already some DDA systems built in, made by a Utrecht University student Pepijn Thijssens, who previously worked on the Futuro Cube. These DDA mechanics focused solely on the FC as a whole, instead of the mini-games you play. This means that the minigames themselves did not really become more challenging, it only felt like they did. I want to change this in this thesis and make the mini-games themselves adapt to the player's skill, making optimal use of the FC's different motions.

While fun and engagement are important topics to research for the eventual goal of making children play more with the FC, this thesis will mainly focus on the training aspect. The goal of this paper is to test out the different DDA mechanics inside the FC and see if it can adjust its difficulty in such a way that children will train and improve their fine motor skills. This thesis therefore proposes the following **research goal**:

The objective of this paper is to analyze how the Futuro Cube minigames can improve the users fine motor skills

In order to achieve this goal, we define two simpler subgoals:

Subgoal I: *What factors should be adapted in a DDA system to change the difficulty of the minigames?*

Subgoal II: *What is the impact of using the Futuro Cube on training the fine motor skills?*

2. Related Work

2.1 Introduction

Looking at different studies over the past few years ([3] and [9]) it has become clear that over the last few years, the number of children having problems with fine motor skills has increased. This increase in fine motor skills problems is something that should be prevented because it can have bad consequences for a child's mental and physical growth.
Worldwide the Movement Assessment Battery for Children Third Edition (MABC-3) is used to determine children's motor development.[10] However this test requires the presence of an expert and also is a long and not very entertaining test. Therefore there has been research in creating smart toys to detect children's motor development. [2] [11] [12] [13] Optimally, we create toys that are fun for the children to detect **and** train their motor mobility.
The Futuro Cube has shown great promise, for it being able to accurately detect fine motor skills. However, the training aspect has not been fully explored. DDA could be a method that could help in this regard. With a well-implemented DDA system, we can make the games slowly grow in difficulty which will make the child slowly improve their fine motor skills in a playful way.

This section will start first by going into more detail about the Futuro Cube (2.2), then explain the principle of flow in games (2.3), then go over the different approaches of DDA (2.4), followed by some examples of rehabilitation games (2.5) and finally go over systems that can help detect fine motor skills delays (2.6).

2.2 Futuro Cube

The Futuro Cube is a 6-sided cube with a 3x3 RGB LED matrix on each side created by Rubiks. It also has a vibration and a sound module. The light, vibration, and sound are used to give visual and audible feedback to the player using it. The FC has a gravity module that allows it to detect which side is facing to the ground and therefore also know which sides are facing to the side and which side is facing up.

The Futuro Cube can very accurately detect small changes to its position. This is very useful for accurately detecting when someone taps one of the sides. When we tap something it every so slightly gets moved based on where we tap it. The future cube is able to record this and evaluate it in the game itself. The precision with which it recognizes movement changes based on tapping allows the FC to very accurately detect even gentle tappings on the cube.

The FC is also good at detecting motion gestures by rotating and tilting the cube around. The FC makes sure that the gestures it can detect by these movements need to be precise in order to not let them happen accidentally when you just are carrying it around and it also does not detect wrong motion patterns while you're actively playing a game on it.

The lights themselves can have different levels of brightness and have a high refresh rate (25 frames per second), which allows them to change colors faster than you can blink. The adjustable brightness also allows different parts of the game to be more noticeable than others.

Lastly, the FC has a radio transceiver built-in that allows the cube to connect to another FC within a short distance of itself. This radio transceiver is included so the user can play games against other players who also have an FC. The radio transceiver makes sure you can only connect to one other FC at a time so that you cannot be interrupted by another player with a FC. The newest FC hardware also allows for connection to your laptop or mobile phone through Bluetooth Low Energy (BLE).

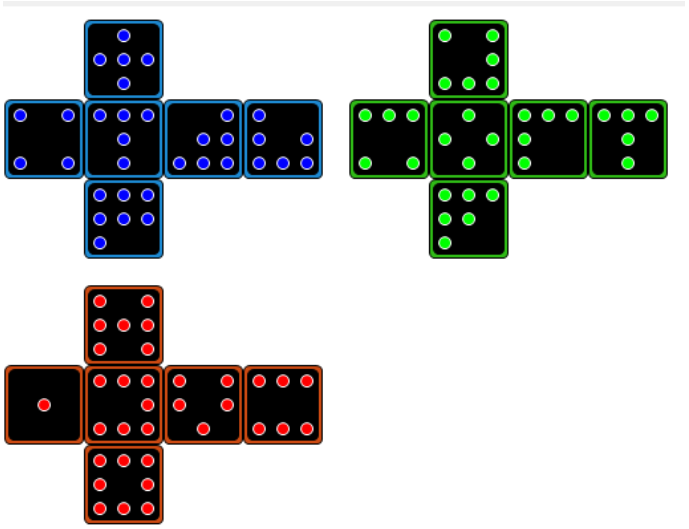


Figure 2.1: The Main Menu of the Futuro Cube

The cube has some basic functionality built-in. To turn the cube on or to return to the main menu from any game we simply make a circle movement in the air. This will open a menu, in which we are able to choose a game. The main menu is a series of ICONS (ICONS are further described in section 2.2.2.2) on each of the sides of the cube. All 6 sides have a different ICON where each ICON represents a game. The game on top is the one that is selected and you can choose to play the game by tapping the top. For ease of access, the FC will also tell the user which game is selected if they tilt to it. If you tap the side of the cube, a new selection of games will show. Tapping twice on the bottom of the cube will turn it off. By default the FC has 3 game menus—including a menu with volume control and a tutorial level—however more menus with self-made games can be added to FC if needed.

2.2.1 Pawn

The Futuro Cube is a small box with many different minigames you can play on it. For this reason, we don't need to use complex programs and systems to have a complete full-scale application written out. The minigames themselves can be written in a simple script for all their functionality. **Pawn** works perfectly for this purpose. Pawn is a simple, C-like programming language, it requires little space and compiles quickly on the spot. More importantly for us, Pawn is a strong 32-bit language that can port from and to the Futuro Cube. Using Pawn we can create small scripts that can be called from within the FC. The Futuro Cube itself has been written in Pawn, therefore there exist already preset built-in functions that can be used.

2.2.2 Hardware

The Futuro Cube V8.14 allows for cooperative multitasking, which means that game scripts and systems scripts can run simultaneously. This way playing music, collecting data, and processing cube tapping or rotating can take place the same time. If one process is required to be finished before another process starts (like music needs to be done playing before you can continue) then you can use Sleep() functions that will halt until all other processes have finished before it starts again. The overall flow of the FC works as follows:

- Wait for the previous process to end
- Start a new process
- If the Futuro Cube detects a new motion it Acknowledges the motion, by making the attached script react based on the detected motion
- whether or not the user made a new motion, it will make the cube draw the new result.

Using pawn we can directly approach any LED by their position or by their index. Their position is a 2-valued index where the first value describes the side of the cube and the second value describes the matrix position (going left to right, top to bottom). Both these values use a 0-index method, i.e. we start counting from 0. This means that the top-left corner of the first face has the coordinates (0, 0). This also means that if you want to target the middle LED of the third face of the cube, you would call the square(2,4). For a clear schematic of the cube for each LED position you can look at Figure 2.3 and Figure 2.4. Another way to call any LED in the code is by using the index of the square. The indexing system uses the same order as the positioning system, which makes it easy to switch between using indexes and square positions in the code.

The FC however does suffer from one small problem: it does not have much memory and it needs to have all the data locally in order to make the games work correctly. So it is not possible to have big scripts saved on the FC at all times. We therefore have 3 separate areas where scripts get stored and loaded.

2.2.2.1 RAM

RAM is the place where we store scripts for sharing data, code, and stack. The main purpose of RAM is for testing purposes. It has a limitation of 11KB space. Because the cube requires some basic functionality to work properly, we have approximately 8kb RAM we can use for our own written scripts. RAM keeps the uploaded scripts in its system until the FC is reconnected through its USB plug. This allows you to test a script even if it is not connected directly to the computer.

2.2.2.2 Flash

Flash is the place where we save all the scripts once they work correctly. Once you have tested the script in the RAM you can upload it to the flash, which can store and run any script you uploaded to it. The Flash can use the 11KB of the RAM if necessary and has its own 50 KB of storage. If the cube is no longer plugged in, then you can play any games based on the ICON created for the game. An ICON is an arrangement of colored lights using the 3x3 LED matrix each side of the cube has. The ICON is displayed on one side and allows you to make different colored forms which helps you separate the different games from one another in the main menu. If no specified ICON was created then a default ICON will be used.

Figure 2.2: Examples of the ICONS for the games, the colors are used to differentiate between the different menu's (see also Figure 2.1).

2.2.2.3 MyCube

MyCube is a specialized area in which you upload personalization code for the cube. You can upload up to 10 KB and use the stack of the RAM to run. You're not able to directly upload scripts to Mycube space. Instead, it is installed automatically if the script's name contains .mycube. ("example.mycube.p")

Scripts don't need to be installed one at a time, it is possible to install multiple scripts at the same time.

2.2.3 Rubik's Futuro Cube Suite

If we want to upload scripts to the FC, we will need to use an additional program: Rubik's Futuro Cube Suite V1.9 (**RFCSuite**) [14]. Uploading games can go from pre-made games (Appendix A.1.2) to games you have completely developed yourself—for example Pepijns Minigames (Appendix A.1.3.1). RFCSuite is further used to update the FC to its newest version if one is available and can be used to update and personalize your user profile on the cube.

2.2.4 code.futurocube.com

We can retrieve active game data from the cube by using https://code.futurocube.com/ [15]. This site is used to upload code, just like RFCSuite. However, this site also allows us to send over the data over Bluetooth, which allows us to keep track of logging information while someone is playing it. Some of the data we can collect this way are:

- how long someone took to complete a minigame.
- how much they move the cube
- when and on which side they press the cube
- how often they fail and succeed
- how far away the player was in completing the minigame (for example, distance from the goal in *Find the dot* (Appendix A.1.3.1)).
- which minigames they played.

We can save all logs locally (in a .txt file) at the end of each test. We can then use the collected data to see if the DDA successfully makes the minigames more difficult.

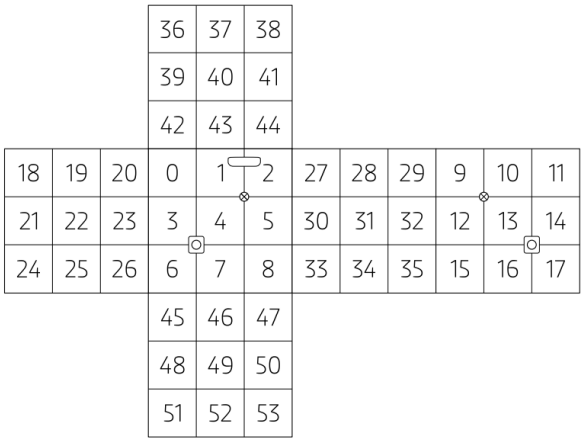


Figure 2.3: Index Schematic of the Futuro Cube

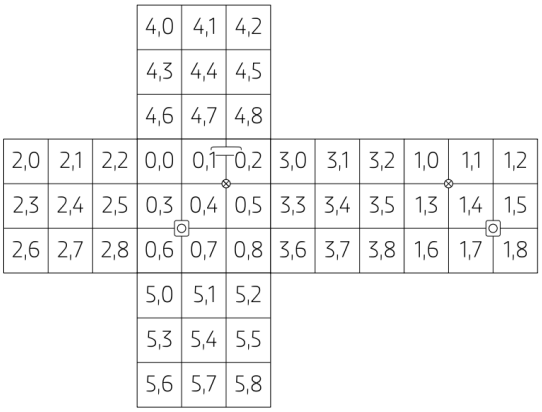


Figure 2.4: Coordinates Schematic of the Futuro Cube

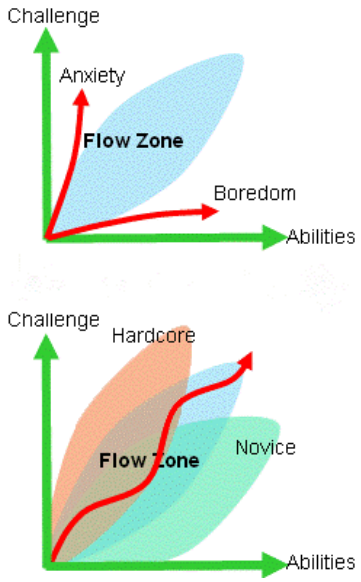
2.3 Flow

Mihaly Csikszentmihalyi, a professor of psychology at Claremont Graduate University, is the first person to describe the concept of flow. You experience flow when you are completely focused on a task and are enjoying said task. When you get completely immersed in the experience, time and other worries disappear while you're in the flow. [16] Flow is also an important part of games. Games have the goal of immersing you in the world that they created, with gameplay and/or with story. Furthermore, player enjoyment is a very important aspect of a game. After all, games are a form of entertainment for a large group of people. Flow is often plotted as Challenge vs a person's ability. Over time, your experience should grow with the challenge you want to overcome. If you become too skillful in a task and it doesn't challenge you in any way from time to time, it will start to bore you and you will stop enjoying it. At the same time, if you are given an impossible challenge that requires more skill than you possess you can become anxious, which also makes you stop enjoying your experience. If, however, the challenge perfectly increases with your skill, you can enter the "flow zone", in which you keep enjoying the experience for a longer time. [17]

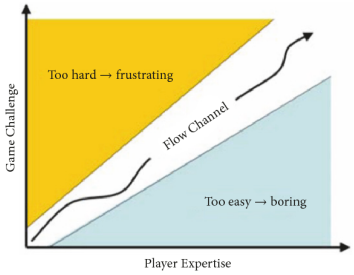
If we look at games as an example, we can see that the player will get the hang of the controls of the game and because of that will increase their skill level. At the same time, the game will start putting up more difficult challenges, for example: increasing the amount of enemies, presenting tougher enemies, giving more complex puzzles, or requiring higher scores. If this is done optimally, people will become immersed in the game and start to enjoy it. This is also what games try to achieve. If it becomes too difficult, it can lead to frustration, because people are unable to continue to make progress in the game. On the other hand, if it becomes too easy, the player can feel bored because they don't feel their skills get properly tested by the game.

Flow does bring one problem with it, and that is that it is unique for every person. Csikszentmihalyi proposed eight components which together make up Flow:

1. A challenge appropriate for a person's skill
2. A merging of action and awareness
3. Clear goals
4. Immediate feedback
5. Concentration of the task
6. Sense of control
7. Loss of self-consciousness
8. Loss or altered sense of time



(a) Csikszentmihalyi flow theory, made by Chen et al. [17]



(b) Game flow chart based on Csikszentmihalyi flow theory, made by Mohammed et al. [18]

Some of these components are not equal for every person. One person (Hardcore, i.e. someone who likes harder challenges) might prefer that the challenges they face become more difficult faster than another person (Novice) who would like to improve their skill for longer before taking on more difficult challenges.

Flow is an important component to consider while working with the Futuro Cube. We want children to continue to play with the FC. Therefore we want them to be engaged with the minigames they play. We therefore will keep the aspects of flow in mind when making new and improving old minigames.

2.4 Dynamic Difficulty Adjustment

Dynamic Difficult Adjustment (DDA) is a technique that the game to change its difficulty while you're playing it, so the player will keep enjoying playing the game. It is meant to keep users inside the flow zone and make the overall experience less frustrating or boring. If it detects that the challenge is too difficult it will make it easier and if it's too easy it will make it harder. DDA is therefore also classified as a dynamic game-balancing technique. According to Andrade et al. [19] a game can only successfully perform dynamic game balancing if the game follows the following 3 requirements:

1. The game is able to quickly detect and adapt its difficulty based on the users' initial skill level. If a new game would be similar to the field someone has expertise in then they will have a higher initial skill level, than a novice or beginner picking up that same game.
2. The game is able to quickly detect the users' improvement/growth or regression in skill
3. The game is able to adapt the game without breaking immersion, i.e. the game adapts the difficulty without the player noticing this change.

If the game is able to do this, then it is also able to effectively implement DDA.

Since 2009 extensive research has been done in the field of DDA, trying to improve this technique either by improving existing techniques or creating new ones. Over the years there have been 7 distinct DDA approaches developed (see Table 2.1)[18]. In the next subsections, we go into more detail for each approach.

Approach	Related Papers
Probabilistic Methods	[20]
Single and multi-layered perceptrons	[21]
Dynamic scripting	[22]
Hamlet System	[23]
Reinforcement Learning	[24]
Upper Confidence Bound for Trees and Artificial Neural Networks	[25]
Self-organizing System and Artificial Neural Networks	[26] [27] [28]

Table 2.1: Table A1: Different DDA Approaches

2.4.1 Probabilistic Methods

Xiu et al. [20] was the first to create a Probabilistic approach for DDA. In some other DDA approaches, we try to predict from moment to moment if the player is in the flow state as described by Csikszentmihalyi. In the probabilistic approach, we try to keep the player in the flow state throughout the entire game. While for different games you require a more complex graph, the basic idea of the Probabilistic graph is as follows (following a typical level-based game):

- The two most important variables are: levels (k) and trials (t)
- There exists a Churn state, which is the state the player will end up in if they decide to quit the game. They can enter this state from any other state.
- From each node the player can enter one of three following states: they stay on the same level (k), but perform an additional trail (t+1); they move to the next level ($k + 1$ and $t = 1$) or they quit and go to the churn state.
- The chance that a player moves to any state can be calculated accurately.

In this model, we can denote the chance that a player will quit after they lost a trial by C^L and the chance they will quit after they won a trial by C^W. Furthermore, we can denote a player's win probability as W_k, t, which states the chance a player will win a specified level given the number of trials they have given it. Using this model, the game can adapt itself accurately throughout the whole game, instead of the greedy approaches that just adapt the game where necessary. It does this by trying to keep C^W low (reducing $W_{k,t}$) and C^L (by increasing $W_{k,t}$). A probabilistic approach has the effect that it will optimize the players' engagement throughout the whole game.

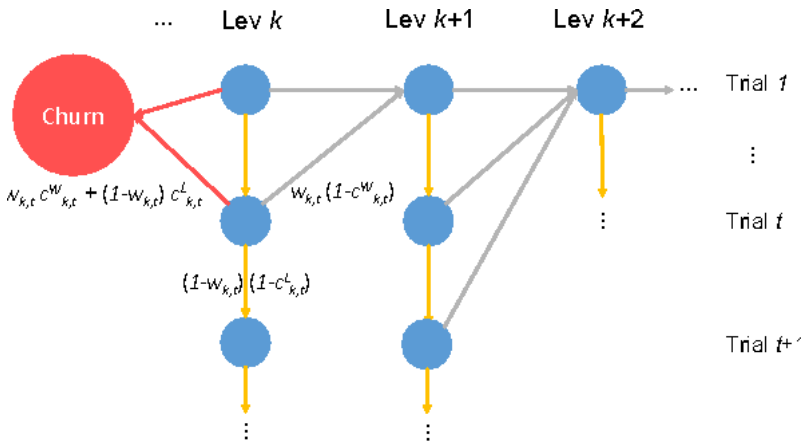


Figure 2.6: Probabilistic graph of a player progression in a typical level-based game, made by Xiu et al. [20]

2.4.2 Single and multi-layered perceptrons

Pedersen et al. [21] performed an investigation to see if it is possible to find a relationship between level design parameters, individual playing characteristics, and player experience (i.e. fun, frustration, and challenge). At first, the authors tried to create a function that would be able to accurately relate a player's experience to the level design and individual playing characteristics. This was not possible for it would result in too much noise. Every person has their own playing style and also mapping emotions to a player's actions is a very subjective measurement. That being said, the authors did find out that a single-layered perceptron artificial Neural Network (ANN) was reasonably precise with detecting both challenge and frustration with players (88,66% and 77.77% respectively). Fun, however, was not quite accurate enough (69.18%), showing that detecting fun in players needs a more complex ANN. If all three factors (challenge, frustration, and fun) are measured adequately we can use the results from it to let the game create the design features themselves. It can detect how the player is doing and create the next part of the game, which is adapted to the player's skill.

2.4.3 Dynamic scripting

Spronck et al.[22] was the first to propose the idea of dynamic scripting as a DDA approach. Every AI-controlled character in the game gets a rulebase, i.e. a set of rules that the AI has to follow. Each of the rules has its own set of weights (which together sum to one) which will help determine what the best action is that the AI can take. The rulebase will execute a specific part of the script, based on the weight values these decisions got. This will make an AI-controlled character perform an action that will either increase or decrease the challenge of the game. The rules that lead to a good challenge for the player will have their weights increase (which decreases the weight of other rules). The rules that lead to the player failing more or lead to the player getting bored by the game, on the other hand, get their weight decreased (which increases the weight of other rules).

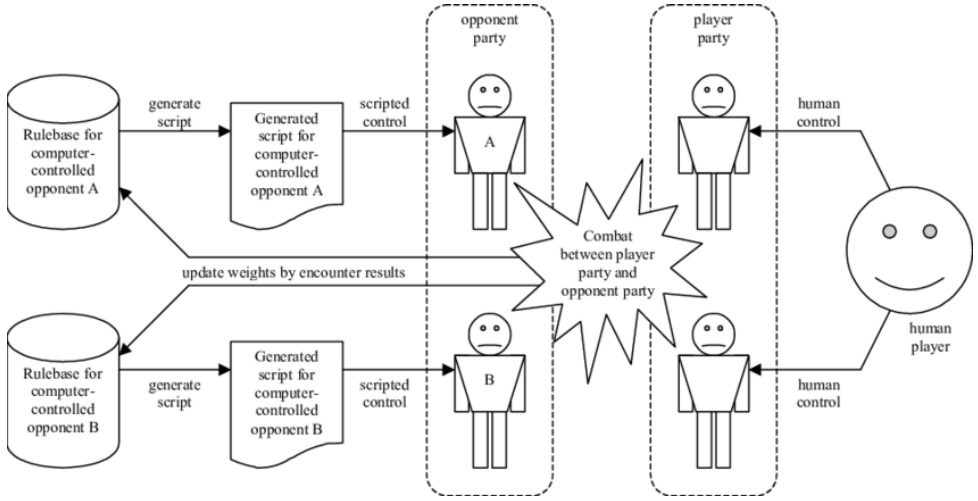


Figure 2.7: The dynamic Scripting process, made by Spronck et al.[22]

The upside of this approach is that the game learns from the player progressively. While you are playing, the game is very good at adapting the difficulty to make it just the right difficulty for you. This does mean that the game needs time to adapt to your skill level and it requires having a good set of rules at the start of the game. If you manage to create good dynamic scripting, however, you can get the additional benefit that the AI seems to adapt its tactics constantly in the game, which would make the player feel more engaged with the game. A downside of using dynamic scripting can be that the game eventually becomes a bit stale. The rulebase will have found an "optimal" playing style and will not deviate from it. This can be solved by one of the following three tactics:

1. high-fitness penalty: one weight being strong—i.e. gets chosen most often—suffers from a higher penalty than having a more balanced weight distribution. This means that a more balanced weight distribution is less likely to change much about their strategy.
2. weight clipping: weights cannot exceed a maximum value, making sure that no weight will be so high that it always gets chosen
3. top culling: Same as weight clipping, however, instead of disallowing weights to become higher than a maximum value, it will just skip these rules if their weights are too high.

All these three tactics allow for a more balanced gameplay loop, where strong and weak tactics are interchanged during the whole game.

2.4.4 Hamlet System

Hunicke and Chapman [23] created a DDA system called Hamlet, which is still actively maintained in the Half-Life engine. Hamlet can detect the player's current statistics and is able to actively adjust the game so that the player stays inside the flow of the game. A simple First Person Shooter (FPS) state machine can be seen in Figure 2.8, this shows what kind of state the player could be in while playing a game. Hamlet can detect the player's ammunition and health state, so if it sees that you are low on ammunition and/or health, it will more likely spawn a room with an ammunition pack or a health pack, and less likely spawn a room with enemies. Furthermore, if Hamlet notices the player is dying often, with the majority of the enemies still alive, it will adjust the game so that it will spawn a health pack inside the room, make the enemy less accurate and strong, and make the player's attacks stronger. If this is still not enough, the game could also actively change the environment, by reducing the amount of enemies, replacing some enemies with weaker ones, and giving the player more ammunition and health packs. Hamlet allows for a clean trial-and-error effect that makes it so that the player never really gets stuck in any one situation. It does suffer from the fact that it can become quite noticeable. If a player starts noticing that the game gets made easier for them, they can feel inadequate. The players will feel like they did not deserve to get to the next point in the game, which could lead to more anxiety in the player. This is very undesirable because it is the thing we wanted to prevent from happening.

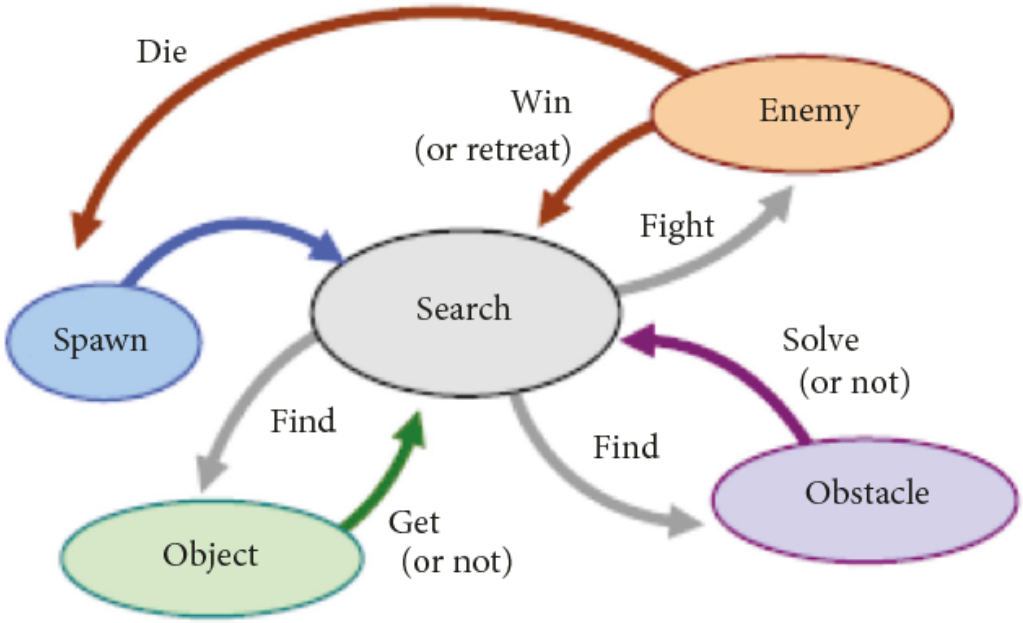


Figure 2.8: Simplified state machine of an FPS game, made by Hunicke et al.[23]

2.4.5 Reinforcement Learning

HagelBack and Johansson [24] found out that players preferred more dynamic AI that was not too easy to beat nor too hard worked best compared to static AI with a default difficulty challenge. An adaptive AI that learns while the player is playing the game is a solution for this. This is a DDA approach that takes place in real-time and is known in other sectors as Reinforcement Learning (RL). The AI gets rewarded on two metrics: Winning percentage difference and mean score difference. The AI learns that his winning percentage (including draws) needs to be minimized, so it needs to minimize $|wins - losses|$ and draws throughout the game. Additionally, the AI learns that the score difference between the players should be as low as possible. This means that for example in a two-player game, $|score_{player1} - score_{player2}|$ should be minimized as well throughout the game. If the system detects that these scores move further from the "optimal" score, then the system will be punished and will adapt itself. If the game on the other hand does come closer to minimizing all the scores, it will get rewarded and learn to perform these actions again in the future. Reinforcement learning does suffer from one thing, and that is that it is a slow process. It will need time to perform a couple of trail runs before it will be able to accurately balance its difficulty with the player's skill, which also constantly is improving and therefore changing.

2.4.6 Upper Confidence Bound for Trees and Artificial Neural Networks

Li et al. [25] was the first to develop an Artificial Neural Network derived from data from the upper confidence bound for trees (UCT). What this entails is that we will explore the part of a decision tree further, when it seems these choices more reliably lead to a positive outcome. The point is that a decision tree can be incredibly large and impossible to traverse in a short amount of time. Using the Monte Carlo Decision Process, we don't know what the future will bring with the choice we make, however, we can predict what will give us the most optimistic outcome and we chose that option. This method is a more offline method and therefore the longer you run it, the more accurate the result will become. It will start to learn which action, based on any state would lead to a good challenge. This UCT can also be performed on an ANN, where the weights and bias get loaded in from a database. This learns quite a lot faster and has a higher win rate than tree-based options. Both these versions suffer from computational expenses though. It costs the system a lot of power to go through trees and see how well it will do with a player's skill. This makes using UCT for DDA systems less favorable than some other options but is still a good option to train the system offline.

2.4.7 Self-organizing System and Artificial Neural Networks

Ebrahimi et al. [29] have a different approach to DDA. They created a self-organizing system (SOS) of Non-Player Characters (NPC) that evolve while the player is playing and adapts accordingly. Using an ANN, the authors are capable of tracking human player traits and use an evolutionary algorithm to modify the ANN weights, which makes the AI adapt to the player's skill.
The best approach for a SOS is to first train the AI system offline, allowing for the of creation of better "parents" rather than just creating completely random parents. Using cross-over and mutation, we let the "best" parents produce new "children". These "children" will then be tested to see how well they perform compared to an average player. From here, the best "children" will be chosen and will become the next set of "parents". If the produced "children" meet a certain requirement set by the creator, then this reproducing cycle will end. Determining the best "parents" is quite arbitrary and is dependent on what you need the NPCs to do. That being said, in general, you want the NPCs to learn that they need to make it challenging for the player, while still allowing the player to win.
After we have created a good set of "children" that can be used against a player, we will order them: from the "child" with the highest fitness (i.e. the one that will have the highest chance of winning against the player) to the "child" with the lowest fitness. When the player starts the game, the median of the list will be loaded in for the NPCs. After a short duration, the system tries to evaluate the human player on the current difficulty and creates a fitness score for it. Based on this fitness score it will change the children that are loaded in that best fit the player.

The result of this is that a SOS is very capable of adapting to many different skill levels of different players. It does come with the cost that it requires a lot of computational power and time to produce these results, which can make it less cost-effective when it comes to bigger games.

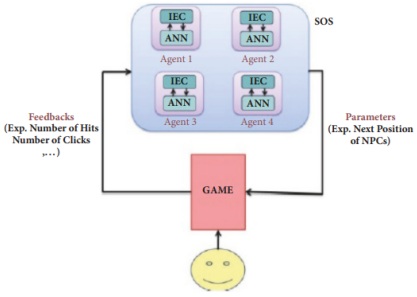


Figure 2.9: Illustration of a Self-Organizing System, made by Ebrahimi et al. [29]

2.4.8 Affective Modeling using EEG

A final approach of DDA is using electroencephalography (EEG) signals to adapt the game so the player will experience the most excitement out of a game. EEG is a noninvasive neuroimaging technique that can detect and read brain signals by tracking electrical signals in the brain. Linking these signals to game events allows the game to produce more exciting (a.k.a. challenging) moments if they detect that the player is getting bored. Contrary, the game will hold on to new events if they detect that the player is getting overstimulated. Using EEG has proven to be very successful in multiple studies and experiments[26] [27] [28], however detecting EEG signals is a difficult process, requiring the player often to be linked with the system, which is not easy to do for every game. Furthermore, these signals are still difficult to detect and interpret, so the system might still miss some signals or misinterpret some signals.

2.4.9 Conclusions

DDA can be approached in many different ways, each having its pros and cons. For this project, we want the minigames to easily adapt and also to quickly adapt themselves to the user's skill level.

The Probabilistic Method works quite well for this. Because the overall goal is to play different minigames that can have clear metrics to define what could make it easier or harder; and clear metrics if the player is succeeding or failing, we can use the probabilistic method to make sure that the chance they stop playing with the FC is as low as possible.

Single and Multi-layer perception approaches or using an artificial neural network are less favorable for it can be quite difficult to detect player experience well with the limited actions it can perform, so creating a complex ANN that can create levels based on a player experience is too complex for our goal.

The Hamlet system is specifically made for the half-life engine, so it cannot be directly used, however, the idea behind it is something we can implement in the minigames. If the player has a very high score we can present them with more difficult challenges and if they are struggling we can give them an easier-to-complete challenge. In a previous research, performed by the UU student Pepijn, this is already implemented partly in the FC.

A basic version of Reinforcement Learning approach could be implemented with the minigames. A positive aspect of using RL for this would be that the cube can "remember" what skill level the user has. Thus if a player decides to play a game on it, it can use this knowledge to set the difficulty just right. The positive effect of this compared to the probabilistic method is thus that we don't have to continuously start from a baseline and try to set the next levels of the minigames based on how the player is performing.

Dynamic scripting and Self-organizing Systems are more methods to make an opponent's AI be of equal skill with the player. However, the goal of this thesis is not to beat an AI, but to let a user improve their fine motor skills. In case we are looking more into implementing games that require an AI opponent to work better, then we can look into using Dynamic scripting or even Self-organizing systems.

EEG is not something that will be used for this project because the impact it would have on the final result is not worth the extra effort it would cost to implement the DDA approach. In future research, people could research if it is possible to implement an EEG system into the Futuro Cube and if it could help improve children's engagement with the minigames.

In Conclusion, the Probabilistic method with some reinforcement learning is probably the best direction to explore for our DDA adaptation. Using the probabilistic method, a good model is created of what would allow players to continue playing the game and to determine which metrics have an influence on it. Then using reinforcement learning, we can teach an AI to generate the best combination of minigames to challenge a player without making it too frustrating.

2.5 Rehabilitation Games

Sometimes a person might lose part of their physical or mental skills. This can be because of a sickness or because of an accident. This doesn't mean that the person will lose these skills permanently. It might be that the sickness gets cured and it will undo most or any of the damage it had done to your physical or mental state. But sometimes—like with an accident—it will not happen on its own, sometimes the body needs to be trained before it can recover to its original state. The way this is done is through rehabilitation. Rehabilitation is a type of therapy to help people regain or improve any lost physical and mental skills. For this literature study, we will mostly focus on the physical rehabilitation aspect.
The problem with rehabilitation however is that it usually requires a long time to recover. Furthermore, you usually need to have a physician to support you. This is both because they have the right equipment for the training and they can direct you if you do some of the exercises incorrectly. It is however not impossible to do these exercises alone and at home. Still, these exercises are boring, repetitive, and might not be challenging enough to alleviate or eliminate the physical problem. Furthermore, you take the risk that a patient will not perform the exercises or perform them badly if there is not an expert to guide them. This could cause however a big workload for doctors and physicians, which is undesirable.
Rehabilitation games are a good alternative. Games are often used as an entertainment platform. Nevertheless, it can also be used for more educative and training aspects. By that metric, a rehabilitation game is also possible to make. The creator needs to talk with experts in the required field about what kind of actions they want the player to perform. These actions would train the player in such a way that would improve their trained skill. Using these games would result in the patient performing their exercises in a more entertaining way. Furthermore, you could allow the game itself to notice if the player performs the exercise incorrectly, relieving the need—and therefore the workload—- of the presence of a physician. Also, doctors need to less regularly check up on the patient and can use in-game statistics to see if they need to schedule an appointment to talk to their patients about their progress.
The next subsections will delve into some example games that have been successfully used for rehabilitation

2.5.1 Upper Limb training

In a paper by Avilez et al.[30], researchers improved their Gesture Therapy system. Gesture Therapy is a low-cost, virtual reality game used to train upper limbs motor skills. The game would simulate normal daily life activities for patients. The patient would be given a small controller that would be linked to a Visual/Sensor tracking module, which would produce a visual gripper in the game. A Database module was part of the system which was able to store the patient's movements while they used the system. The problem with this system is that a therapist was required to be present at all times, to make sure that the challenges the game would give the patients correlated with their rehabilitation process. This also means that a patient could not just go and train further at home and always needed to go to the rehabilitation center to train their upper-limbs motor skills. The authors of this paper therefore added an adaptation module to the Gesture Therapy system, to solve these issues. The adaptation module is a Partially Observable Markov Decision Process (POMDP) that can accurately determine if the challenge the game gives the patient is too hard or too easy. It does this with the help of two hidden variables: performance and difficulty. The difficulty has three levels: easy, medium, and hard. The performance has also three levels: bad, good, and outstanding; and is determined by the observable measures: control and speed. Control is determined by how straight the user can move the gripper from their original position to the goal position. Speed is determined by the execution time of the user over the path they traveled to the goal position. Control and Speed are both measured in three ranges: low, normal, and good. The difficulty is directly related to the performance of the patient. If the patient shows bad performance then the difficulty is set to easy; if the patient shows good performance then the difficulty is set to medium; and if the patient shows outstanding performance then the difficulty is set to hard. The system does this by giving one of three actions: increase the difficulty by one level, decrease the difficulty by one level, or keep the current difficulty.

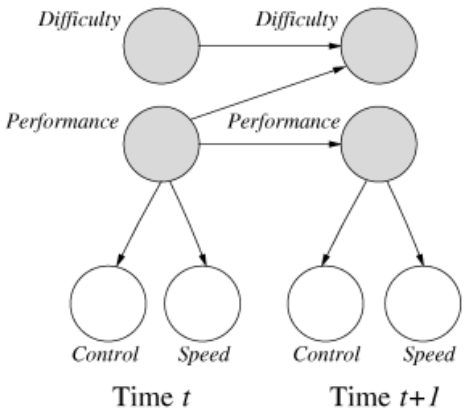


Figure 2.10: The POMPD of the adaptation module of the Gesture Theory system, made by Avilez et al.[30]

2.5.2 Posture Training

Pirovano et al.[31] made a new rehabilitation game that is both able to monitor the patient's movement and adapt to the patient's rehabilitation status. Pirovano et al. recognized that during the rehabilitation process patients usually start recovering and will require the game to be able to adapt in real-time to the patient's status. The game is capable of performing two separate adaption methods. The first method is the game allowing itself to change the difficulty—within some constraints set by the therapist—based on the patient's performance. The second method is that the therapist is allowed to reconfigure the game if they feel the need for it.

(a) Fuzzy Severity levels, made by Pirovano et al.[31]

(b) Example of the system giving the user an error-level warning, made by Pirovano et al.[31]

One of the games the authors made with the clinic experts for this research was a game called Fruit Catcher. Fruit would fall from a tree within a certain range (determined by the therapists). They could also determine the size of the fruit and how fast the fruit will fall (based on normal gravity stated as g). The goal of the patient would be to catch as many fruits as they can by moving their character using an input device (in this game the Wii Balance board was used). To prevent the patient from "cheating" the game, the therapist is also able to set some configurations that would limit how far the patient would be able to step off the Wii balancing board for example. They could also set some constraints that would limit the patient from moving their head too much if they want to prevent the patient from constraining parts of their upper body too much. The game would use these constraints in fuzzy-based monitoring to see what actions needed to be taken:

- **silent**: No action is taken
- **log**: No visual action is taken, but the system logs the risky movement so that it can be analyzed by the clinicians.
- **warning**: If the user breaks one of the constraints set by the therapists, it can trigger a warning. It will tell the patient to correct their bad behavior and log the event.
- **error**: If the user breaks the constraint in such a way that it could be considered dangerous for the patient—i.e.the behavior could lead to the patient getting worse in his or her condition—the game will pause and do the same things as warning level. The system will tell what the patient did wrong and log the event. Afterward, the patient can choose to restart or resume the game they are playing.
- **shutdown**: This level is only triggered if the patient caused some movement that could be considered highly dangerous. It will immediately stop the game and send a message to the clinician to seek immediate contact with their patient to see if they are alright.

This monitoring is also used for the DDA system attached to it. If the game has to log barely to no messages and the game notices that you're very successful in the game, for example by catching the fruit consistently, it will increase the difficulty. On the other hand, if the system needs to send out a warning level alarm or higher, it will decrease the difficulty because it's clear that the player is trying to compensate for the fact they are failing.

2.5.3 Conclusion

These two examples show that games are an effective method to reduce the pressure on clinical experts. Games are a great way to keep the patient engaged, while they still train their motor skills. Additionally, games can easily be made flexible in a way that they adapt to each patient's personal needs. We even can go as far as letting the game adjust itself in real-time, making sure that every patient optimally trains their motor abilities. If done correctly, we know that the Futuro Cube could also have the potential of using its minigames to train the fine motor skills of children.

2.6 Detecting delayed motor skills development of children

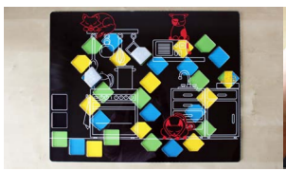
If someone suffers from an illness then it is not necessarily impossible that they will one day recover from it. Though, preventing sickness is of course better than curing it. This is also the case when it comes to motor skills development. It is better to prevent children from suffering from delayed motor skills than to train these skills because they are so underdeveloped.

Worldwide— including the Netherlands —the Movement Assessment Battery for Children (MABC) test is used to assess children's motor development. Over the last few years mostly the second edition (MABC-2 [8]) has been used, however, since Winter 2023 the third edition has been released (MABC-3 [10]). Still, because this edition is new, most literature will not have had a chance to use it. Additionally, a Dutch version is yet to be released as of September 22, 2024. Therefore, we will mainly focus on MABC-2. The MABC-2 test is a two-part test that is used to test a child's motor skills. The first part of the test is a checklist, which is to get a general baseline of the child's skills. The second part is a motor test in which the child's balance, hand coordination, and ball coordination get tested by an expert. Afterward, the expert can hand out a score on each part of the test, in which they can assess if the child suffers from or runs the risk of developing delayed fine motor skills. While MABC-2 can quickly and accurately determine if a child suffers from delayed fine motor skills or has a risk of suffering from it in the future, it can be a quite boring test for the children and it requires a trained supervisor to be present. This trained supervisor can also not be just a teacher who teaches the children. It needs to be someone who has followed a workshop so that they know what they need to look at to asses a child's motor skill development. Even if a teacher would have followed such a workshop, it would still require the teacher too much time to be able to accurately test all their pupils. Sensor-augmented toys are a possible solution to this problem. It has been determined that sensor-augmented toys can predict the outcome of a MABC assessment. [13].

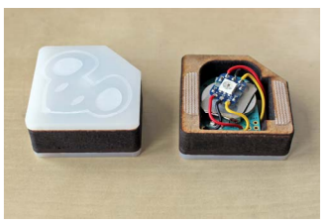
In the following subsections, some examples are shown of sensor-augmented toys in an attempt to accurately detect the motor skill development of children.

2.6.1 Mouse Tokens

Mironcika et al. [12] created a small sensor-augment token together with an interactive board. The board depicted a kitchen with sleeping cats on it and colored slots in which your token would fit. The token was a 38mmx38mmx18mm quadrant with a cut-off corner, a built-in accelerometer, and two RGB LEDs. The faces of the token were transparent and had a mouse logo on it. The token represented a mouse and each player would get one of these tokens. The goal of the game would be to get your mouse the quickest to the other side of the kitchen without waking up the sleeping cats. The way they could achieve this was by flipping the token with one hand en seeing what color lights up at the bottom of the token (green, yellow, or blue, which corresponds to the slots on the board). If you flip the token too fast, you wake up the cat. The mouse token would show a red color and you needed to return the token to its original location. The goal of this board game was to create a game where the children could pick up the token, flip it around while holding it in one hand, and precisely place it into a designated place. Furthermore, the game should encourage smooth hand movements and the game should adapt to each child's motor skills so that it is enjoyable for everyone.



(a) The board of the game, depicting the holes you could place your token in and the sleeping cats in the kitchen.



(b) The mouse token, created by Mironcika et al.[30]

Out of the user study, some points of feedback returned the following:

- While picking up and placing the mouse tokens were very easy actions to perform, flipping with one hand caused some problems and resulted in children compensating it through other methods. This caused the required motor skills to be trained or tested incorrectly.
- The token was a little bit too big for comfortable manipulation with a single hand.
- The difficulty adjustment added to the game could be improved because it was still too difficult for some children
- The competitive side of the game (playing against other children), resulted in children being distracted from the original goal: testing their motor skills. This also resulted in the fact that some children would overcompensate in order to win. This overcompensation would result in either the user "cheating" the game or using their other hand to help complete the hand motion. A possible solution for this could be if the game would change to a cooperative game, but it would require a new user study to confirm this.

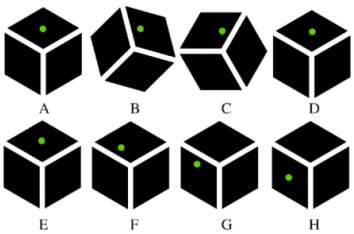
In the end, the authors concluded that the mouse tokens were unsuccessful in accurately detecting delayed fine motor skills of children. But, it did still show that it is possible to create board games that could detect these issues. Working further on this board game or creating a new sensor-augmented toy could result in one that is capable of detecting and training fine motor skills.

2.6.2 Futuro Cube

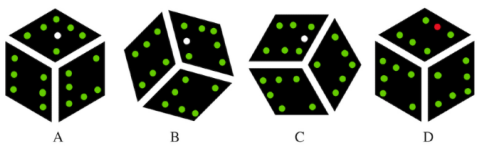
Brons et al. [2] used the Futuro Cube [1] to see if sensor-augmented toys could detect delayed fine motor skills in children. In the paper, they let the children play with two different minigames that could be played on the Futuro Cube: RoadRunner and Maze.

- In RoadRunner a dot on the Futuro Cube would appear and would move in a random direction. The goal would be to keep the dot at the top face of the Futuro Cube by moving the cube the opposite way as the dot is moving to make it move back towards the middle of the face. A predetermined difficulty (easy, normal, and hard) could be set which determines how fast the dot would be moving around.
- In Maze a white dot would show and a small maze would be created existing out of green dots. The player can move the dot around by rotating the Futuro Cube in that direction. If the player hits the wall of the maze (green dot) then the wall turns red and the player needs to move the dot back to just before they hit the wall. The goal of the game is to move the dot through the maze. A predetermined difficulty (easy or hard) could be set, to determine how difficult the generated maze should be.

After the user studies it became clear that the Futuro Cube and possibly other sensor-augmented toys are capable of detecting a child's fine motor skills. Furthermore, the Futuro Cube showed that sensor-augmented toys can predict the MABC-2 test with a 70-75% accuracy. The minigames tested also showed that games focusing on speed are better at predicting the motor skill level of the user than games focusing on precision.



(a) schematic of the game Roadrunner, made by Brons et al. [2] Pictures A-D show what the player is supposed to do to keep the dot in the right place and E-H shows what happens if the player fails.



(b) schematic of the game Maze, made by Brons et al.[2]

2.6.3 Conclusion

The Mouse tokens were a good example that helped us narrow down what we should avoid with our research. To start of, we should focus on creating movements that use both hands. We saw with the mouse tokens that using one hand resulted in overcompensating with the other hand. By making sure the users use both hands, we can make sure both get trained and not just their dominant hand. Also, we know that we should try to avoid the competitive side of the minigames. The goal is to let the users train their fine motor skills, not to become the best in the minigame. If we would have two players try to beat the same minigame after one another to see who did it better or if we were to create a global leaderboard where players would be able to compete against one another, then the players most likely lose track of the original objective. They will start trying to become the best in that specific minigame, which will not train them in a variety of different hand gestures, and therefore will not train their fine motor skills well. An additional negative effect it could have is that people with delayed motor skills will stop playing with the minigames or with the sensor-augmented toy because they feel like it's impossible for them to ever be as good as the top players. A personal leaderboard on the other hand could possibly help, because it can give active feedback showing that the player is improving, which could motivate them to continue using these sensor-augmented toys. The Futuro Cube has already shown that it was capable of predicting the MABC-2 test and it was able to accurately detect a child's fine motor skills. With this knowledge, we can try to use these predictions for a DDA system in our project. Because we know the Futuro Cube is accurate in detecting the fine motor skills of children, we can also use it to see if it improves it. If the data shows improvements in fine motor skills as children use it for a longer period, then we know that the Futuro Cube can also be used for training and rehabilitation purposes.

3. Methodology

3.1 The MinigameSeries

This thesis aims to train children's fine motor skills through playing with the Futuro Cube. For this goal, we have implemented a minigame series. This is a single game where different kinds of smaller minigames are randomly picked for the user to play. For more detail about this see section A.1.3.1. The original minigames are implemented by Pepijn and are changed by Maik. Maik also added the new minigames to the minigame series. On the adaptations made on Pepijn's Minigames see the section below: Futuro Cube adaptation (3.5). To easier differentiate between the different versions, we will from here on out describe the different versions as MinigameSeries.
MinigameSerie1 is *Pepijns minigames*, this is the original set of minigames of Find The Dot, Tap Side, Spinning, and Shaking.
MinigameSerie2 is the same set of minigames as MinigameSerie1, but with the adaptations made by Maik. These adaptations are described in more detail in section 3.5.1.
MinigameSeries3 is the full new version that Maik created from MinigameSerie1. This includes all the changes to the original set of minigames, like in MinigameSerie2, and includes the new minigames added to the game. All changes made to this version are described in section 3.5

3.2 Retrieving the data

The FC can record and save high-score data of the minigames you play, however, this data is very limited and can make it difficult to know what the score represents compared to the player's skills. Game data, like how long a player takes to complete a minigame, how far off it was on completing a minigame, which difficulty level the game was on, what the randomized outputs were, etc. is much more useful information to have. There are therefore also some ways to retrieve this game data from the FC directly. First off, we have RFSuite, in which you can print any game data directly to a console. This does bring 2 problems with it:

1. The FC needs to be connected to a laptop/computer through a micro-USB cable. This is a problem because the cable will be in your way and will limit the motions you can perform with it. Also without a long micro-USB cable, it can easily result in the cable getting launched out, which breaks the connection and prevents any data from getting transmitted
2. The FC has a limited RAM. MinigameSerie1 is already slightly too large for the FC to handle with only its RAM. The consequence is that we cannot add new minigames or adapt the existing minigames without causing the RAM to overflow and the cube to crash when we start the game.

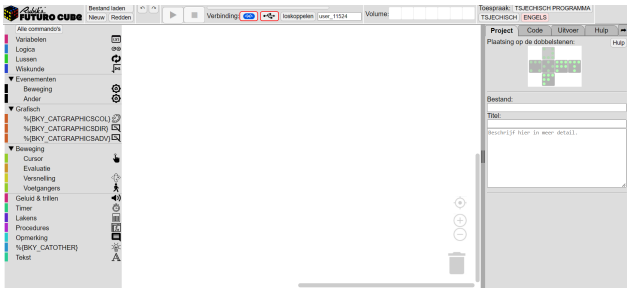


Figure 3.1: A screenshot from the : https://code.futurocube.com site (translated to Dutch through automatic Google Translate functionality, for the site itself is in Czech)

A way to solve the cable problem is to instead send the data over through Bluetooth. The newest firmware of the Futuro Cube makes use of Bluetooth Low Energy (BLE). Using BLE you're able to connect with it through other devices (i.e. laptop/mobile). For the laptop, the following site exists that can be used for this: https://code.futurocube.com/ [15]. This is a site that allows you to write and upload code over BLE through a web API. Runnable code also puts out any desired game data to the output console present in the API. However, this website still suffers from the RAM limitations problem due to it uploading the code the same way as RFSuite with a micro-USB cable does. There is however another app/program we can use to retrieve data from the FC using BLE: **nRF Connect** (IOS, Apple, Windows) [32]. nRF Connect allows us to retrieve data directly from the FC while someone is playing a game on it. This removes the barrier of first uploading the entire program file to the limited RAM space. Instead, we can use the RFSuite Cube Manager to directly install the script on the FC Flash (which can store more data on it). Then we can use the nRF Connect app to connect with the cube and tell it to send over all game data it receives. The sent-over game data is the same game data that you would receive from using the output console on the other two methods.

3.3 Experiment with the Experts

While the data is useful, it is not the only assessment criterion to determine the feasibility of using the FC to train delayed fine motor skills. We also want to talk to an expert to check what they would want to see improved with the FC. The goal of this thesis is to find out in what way we can change the FC so it will help train users in their fine motor skills. The data can guide us on what works and doesn't work, but experts will have the best idea of what would help in achieving our goal in the long run. dr. Annette Brons, a researcher who worked with the FC before, provided some contact information for a couple of experts who could provide their insight on the FC. One of these experts is dr. Daniel Bossen, an exercise therapist who uses new technology to make people exercise more and eat healthier. Daniel is also connected to therapists who often use technology for rehabilitation and training. These experts are first given a series of small tutorials, to make sure they understand how both the minigames and the FC work. For each minigame, a tutorial is made in which the expert can try out the minigame for as long as they want. First, the minigame is explained. At the same time, it is shown how to complete the minigame. Afterward, they are given the FC to try out the minigame. Here they are able to test what happens if you fail, what happens if you succeed, and if you can succeed in a certain way. Afterward, they answer some questions to give an overall idea of what they think works well for the FC and what needs to be improved further. The goal of this questionnaire is to get a good idea of what the experts believe works well and what aspects are still lacking. The games and the correlated motions that work well can be looked into to see why they succeed in training the user's fine motor skills. At the same time, we can look at why other games are failing to achieve this goal. Furthermore, the experts tell if certain motions are still missing for good overall training of someone's motor skills. For example, a certain motion might not get trained/used enough or a motion is completely missing.

Using the feedback from the experts we adapt the Futuro Cube to change the minigames or add new minigames. Once these minigames have been adapted and tested, we go back to the experts one last time to see what their new opinions are on the changed Futuro Cube. The questions revolve mostly around the fact whether they believe the adapted FC can be considered improved since the last time they used it. The DDA system is also built-in at this point, so there are also some questions about this. They are asked if they notice the FC adapting to their skill level, or if the FC is making it too hard or too easy.

3.4 Experiment with Students

The experts help give great insight in what would work and what wouldn't work, but seeing it in action gives more clear results. Therefore we want to also let students/alumni use the FC to see if it does its job correctly. In ideal circumstances we would let small children use the FC for a longer period of time, however, due to the time constraint and the additional privacy concerns attached to it, we have used students/alumni as substitutes. This experiment is mostly used for the first part of the research question: *What factors should be adapted in a DDA system to change de difficulty of the minigames?* The students are first asked a series of personal questions (all gathered data is anonymized afterwards), to see if any environmental or genetical circumstances could have any influence on the test results. Then they are given the same tutorial the Experts were given to show off the different minigames and to get acquainted with using the Futuro Cube for the minigames. If they feel ready, they are allowed to perform a test round with the minigame series. They get to play it out and see how well they perform. After they have done it once and think they understand how it works, they are asked to play it one additional time, this time while their data gets recorded (This data also gets anonymized). After they have completed the minigame series twice over they are asked a couple of follow-up questions to ask how well the flow of the game felt (i.e., how difficult was it to complete, how frustrating were the minigames, how boring were the minigames). This was done using the Flow Short Scale of Rheinberg et al[33]. Here the first 10 questions are questions about Flow, so how much they get absorbed playing the game, forgetting what happens around them, and how natural all the actions come to participants while they are playing the game. The last 3 questions are classified as Worry, which is used to determine any potential Risks the activity (in this case, the game) could bring. These Worry Questions are to see if the participants experience any form of anxiety while playing the game, which is something we want to prevent. After they have filled in the questionnaire, they are rewarded with a self-made cookie as thanks for participating in the experiment.

3.5 Futuro Cube adaptation

3.5.1 Adaptations to the existing games

To make MinigameSeries2, some adaptations were made to the original games on MinigameSeries1.

- Find the dot: the dots the user has to find spawn further away and the user needs to find multiple dots to complete the minigame.
- Tap: To make it more difficult, more sides need to be pressed.
- Shake: the amount you need to shake is changed based on the difficulty level and also how long the user has to shake the required amount.
- Rotate: on higher difficulty levels, the user has to make more steps and has less time to perform the required amount of steps.

3.5.2 Remade Roadrunner

The game *Roadrunner* (A.1.1.3) is a standard game on the cube that would require some good use of fine motor skills to play properly. Additionally, in Appendix A we described how this game could easily be adapted in different ways to make it harder or easier. Therefore MinigameSeries3 has been expanded to include this game. The built-in DDA system has been adapted to also work correctly with the addition of Roadrunner.

3.5.3 New Snake Version

The Game *Snake* (A.1.1.11) is another game that is by default on the cube. It requires some amount of fine motor skills to handle to make the snake move the correct way. At first, the snake game was not added due to the space limitation of the FC and the limiting aspect of using fine motor

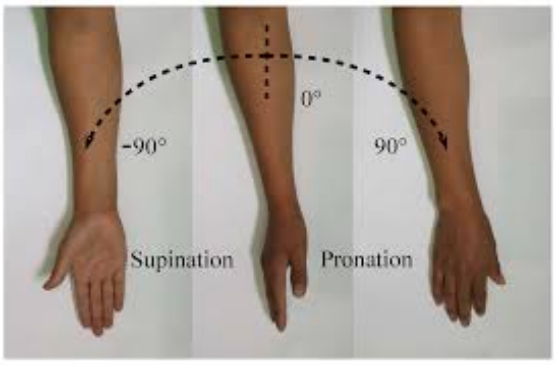


Figure 3.2: Visualisation of the Pronation and Supination motion. Rotating your arm from palm up to palm down (left to right) is called pronation. Rotating your arm from palm down to palm up (right to left) is called supination.

skills. However, after performing the experiment with the experts, a suggestion was made by them to have an adapted version of Roadrunner in which you could change direction by tapping the side on the cube and that your goal would be to collect dots. This was very similar to what you do in Snake. So, in the newer version of the FC, Snake was added to MinigameSerie3. However, compared to the standard version of Snake the FC has (also described in A), this snake works a bit differently. This new Snake version uses taps on the side of the cube to make it turn, instead of turning based on the direction the user is holding it. The poisonous apples have also been removed from the game, in order to focus on the player's fine motor skill over their puzzle-solving skills.

3.5.4 New Game: Flip

One additional motion was recommended from the experiment with the experts: Pronation/Supination. This is the motion of rotating your arm, from palm up to palm down (pronation) or vice versa (supination) (see also Figure 3.2 for a visualisation of this motion). We achieved this new motion with a simple new minigame we call Flip. The FC can detect which side is the face on top. Additionally, each face has its own designated number, so we can easily retrieve which side is the opposite side of the face that is currently up. With these two pieces of information, we just check if the player manages to correctly turn around the FC to its opposite side in a given amount of time. We can also increase the difficulty of this minigame by increasing the amount of times you have to flip the cube and how fast you have to turn it to its opposite site before it resets itself.

3.5.5 New Game: Special Shake

A last feedback point from the experiment with the experts was making an adaptation or improvement of the Shake minigame. According to the experts, the motion you have to make for the shake minigame cannot be considered fine motor skills for it requires your whole arm to move for it. To try to combat this issue an alternative version was made of this shake minigame, called Special Shake. Instead of asking the user to shake the whole cube, we require the user to shake the cube in a specific direction, thereby forcing the player to make smaller arm movements and therefore train more their fine motor skills.

3.5.6 The Consideration Maze

The game *Maze* (A.1.3.2) was also considered to be added to MinigameSerie3. However, with the added 4 new minigames additional to the already existing 4 minigames, it was eventually not added to prevent too much overhead. It was already clear from a couple of test runs, also with the Experts, that it would become very difficult to keep track on what minigame was currently active and what you would need to do for that minigame. Additionally, compared to most of the other minigames, Maze requires slow and precise movement. A sudden switch to this minigame after more high-speed minigames like *FindTheDot*, *Rotate* or *Roadrunner* could mean that the user fails the *Maze* minigame before they even had started it.

3.5.7 Added Sound

After the experiment with the Experts it became clear that, even after a short tutorial and explanation, it could be difficult to understand what you were supposed to do while you played the game. To this end, custom sounds would be added to announce what minigame was chosen and what you're supposed to do in the minigame. For example, for the Tap minigame the FC would play a sound that would say "Tap the flashing side", making it clear to the player they were supposed to tap the side on the FC that started to flash at that moment. For *FindTheDot*, *RoadRunner* and *Snake* it is not possible to quickly announce within a second what you need to do, so for these minigames the name of the minigame would only be played and the user would be required to remember what these minigames exactly entailed.
The custom announcement sounds were made with Elevenlabs[34], using the *American Voice of Liam* for its voice being a close match to the original voice lines already present in the FC. The only problem with the files Elevenlabs produced is that they are mono 16-bit sample width WAV files with a 44.1 kHz sample rate, while the Futuro Cube can only handle mono 8-bit sample width WAV files with a 22.05 kHz sample rate. To fix this issue, we used Audacity 3.5.1[35] to format the sound files to the required format. Additionally, Audacity would be used to increase the volume of the WAV files, so the sound would be balanced with the other music files used during the game.

Apart from adding and using the custom announcement sounds, a delay was implemented in the new version of the FC. This one second delay would automatically take place when the game switches between minigames, so the user has a second to recognize and recollect what they need to do for the minigame that is shown. It would also give them some time to see that the minigame had switched to a new minigame.

3.5.8 Old Version Bug

A strange consistent bug occurred, which has an unknown cause. Whenever on the old version we would retrieve data through nRFConnect, on the minigame Shake it almost always completely crashed the game and the FC as a whole. The FC would then become unresponsive to any action you would perform and you would need to connect it with a micro-USB cable to your laptop to get the FC back to its original state. The cause of this bug was eventually found to be the sound module, though the reason for it is still unknown. If we were to do the game without collecting data using nRFConnect, it would work fine, also if it was played with sound. If we were to use the new version, which has the exact same code, and retrieve data using nRFConnect,

it would work fine, even if we play it with sound. It is specifically if we play sound effects on the Shake Minigame in the old version of the game and retrieve data from it with nRFConnect, then it had a high likelihood to completely break the FC. To solve this issue and still be able to retrieve the required data from it, the sound effects of the minigame Shake were removed from the old version of the game. The participants were also told about this issue and that they should not expect sound from that specific part of the experiment. The removal of the sound has been assumed to have minimal effect on the end result based on the participant's performances.

3.5.9 Adaptations

Currently, there are 2 present DDA systems built into the minigames system:

- The time you have to complete a minigame: if you fail the minigame the amount of time you have gets increased, while if you succeed it looks at how much time you have left and reduces the time you have for the next minigame. The more time you have left over, the more time is reduced on the next minigame.
- Some minigames are considered "more difficult". For example "spinning", is considered more difficult than "Tap side" (see A.1.3.1). If you have failed too many minigames, the easier minigames are chosen over the harder ones.

However, we have added a new DDA system: making the games themselves harder or easier. For the 4 minigames—MinigameSerie2 of *Finding the dot*, *Tap Side*, *Spinning* and *Shaking*—we made some changes to make them more adaptable. We base this system on an adaptability variable (AV) with a scale of 1-10, where 1 means that the minigame is made very easy and 10 means that the minigame is still doable, but very strict in the way you're supposed to complete it. This scale is from here on out referred to as different AV levels. See the section 3.5.10 below how these AV levels change between minigames. The changes that are made for MinigameSerie2 for this new DDA system are as follows:

- *Finding the dot*: instead of finding an empty spot on the cube, we find a random spot on the cube that is a specified distance away (where the AV determines the distance). On higher AV levels (>= 8), we also increase the amount of dots you have to find in quick succession within the time limit.
- *Shaking*: depending on the AV level, this minigame will require the user to shake longer/shorter for some duration. Furthermore, this minigame will also require you to shake more/less aggressively (i.e. shake more/less in the same amount of time). for AV levels of 3 or lower, you have a long amount of time to not have to shake as much; For AV levels 4-5 we have around the normal amount of shakes required; For AV levels 6-7 you need to shake it for longer period of time and for AV levels 8 or higher we have that you have to shake it for the same amount of time as for AV level 7, but you are also required to have shaken more in that same time-span.
- *Tap Side*: Two changes have been made to this minigame. The first one is that it sometimes requires more than 1 tap for it to register as a successful minigame. The amount of taps required is dependent on the AV level. The second change is that the sides you need to tap can randomly change during the minigame, sometimes requiring you to quickly rotate the cube. This second change only happens on higher (>= 7) AV levels.
- *Spinning*: Spinning has two small changes added to it. The first one is that the AV level determines how much the cube should be rotated around. The second change is the speed in which you need to rotate it (the time you have to complete the minigame) is increased/reduced based on your AV Level. This time is only adapted if the AV level is 3 or lower (for increasing) and 8 or higher (for decreasing)

Table 3.2 also shows the parameter values used for each AV level for MinigameSerie2.

For newly added games to the MinigameSerie3, we have added a DDA system as follows:

- *Roadrunner*: For Roadrunner the starting speed, how often it will change direction, and how

quickly its speed increases are all determined by the AV level.

- *Flip*: For the Flip minigame we determine how fast you need to flip the FC and how often you have to flip it in the given amount of time. The higher the AV level, the more you have to flip and the faster you have to flip it.
- *Snake*: For Snake, we increase the walking speed of the snake, the amount of apples you have to eat, and how long your tail is when you start. The last one is significant because it makes it harder to move around. After all, if you hit yourself, you automatically lose the minigame. The higher the AV level, the longer the tail, the more apples you have to eat, and the faster you move.

Table 3.3 also shows the parameter values used for each AV level for the new minigames.

3.5.10 Scaling the Adaptability Variable Level

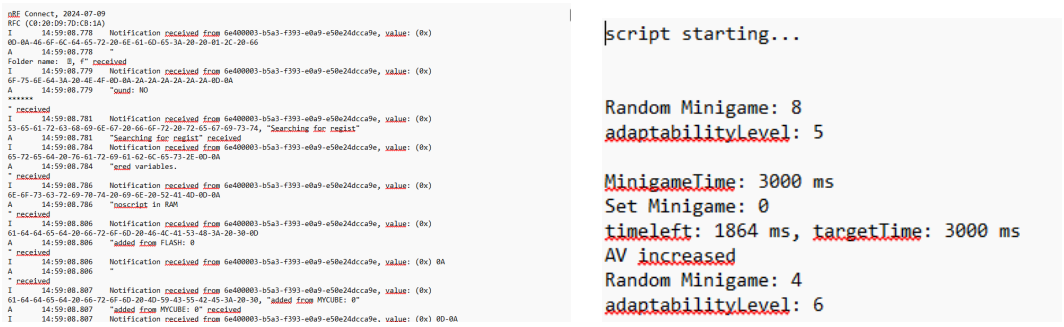
During a singular game, a total of 9x6=54 minigames need to be completed. Taking into account that the player will probably not be able to complete all minigames in one go, an estimate of about 60 minigames will at least be played. Throughout these 60 minigames we want the AV level to be around the player's skill level. For the first 10 minigames, the AV level will change more drastically. Increasing by 1 for every minigame you manage to successfully complete and decreasing by 1 for every minigame you fail. After that, the next 10 minigames increase the AV level if you succeed in 3 minigames in a row, and decrease the AV level if you fail 1 minigame. From there the next minigames increase the AV level if you succeed in 5 consecutive minigames and decrease the AV level if you fail 3 consecutive minigames. After completing about 50 minigames, it is assumed the user is at the preferred AV level and will no longer change. Table 3.1 gives a visual representation of this. Additionally, Algorithm 1 below shows the pseudo-code of the used algorithm to change the AV level.

Minigame Number	Decrease level?	Increase Level?
0-10	1 level failed	1 level succeeded
11-20	1 level failed	3 consecutive levels succeed
>21	3 consecutive levels failed	5 consecutive levels succeed

Table 3.1: Scaling of the AV level over the course of the minigames that get played. The minigame number describes how many minigames have been played (both including successful and failed minigames). "Decrease Level?" shows how many levels in a row have failed to reduce the AV level by one and "Increase Level?" shows how many levels in a row have to succeed to increase the AV level by one.

3.6 Preprocessing the data

The data we receive from nRF Connect is not all useful to use. This is because the nRF Connect app also includes the exact time when it sends and receives notifications, the device where it comes from, and the type of notification it receives. To make processing the data easier, some solutions were written to filter out most of the unimportant data. Figure 3.3 shows an example of how the data is saved during the experiment (left) and what it looks like after it has been preprocessed (right). This data we then can easily convert into an Excel file. To make that process faster and to make sure no data gets lost, some additional solutions were written that can set the .txt files over into understandable Excel files.



(a) Unprocessed data **(b)** Processed data

Figure 3.3: Example of gathered data in a .txt file before and after processing it

Algorithm 1 Adaptability Algorithm

```
1:  procedure UPDATEADAPTABILITY
2:      AV ← current AV level
3:      MinigameNumber ← Amount of minigames played
4:      Fails ← How many minigames have failed in a row before the current minigame
5:      Successes ← How many minigames have succeeded in a row before the current minigame
6:      isSuccessful ← whether or not the current minigame was successful
7:      if isSuccesful then
8:          Successes ← Successes + 1
9:          Fails ← 0
10:         if MinigameNumber < 10 then
11:             AV ← AV + 1
12:         if MinigameNumber < 20 and Successes = 3 then
13:             AV ← AV + 1
14:             Successes ← 0
15:         if Successes = 5 then
16:             AV ← AV + 1
17:             Successe ← 0
18:         if AV > 10 then
19:             AV ← 10
20:     else
21:         Fails ← Fails + 1
22:         Successes ← 0
23:         if MinigameNumber < 20 then
24:             AV ← AV − 1
25:         if Fails = 3 then
26:             AV ← AV − 1
27:             Fails ← 0
28:         if AV <= 0 then
29:             AV ← 1
```

AV Level	Finding the dot	Tap Side	Shaking	Spinning
1	max distance: 1 dot amount: 1 time: 5s	amount of taps: 1 time: 5s	Shake-amount: 250 time: 8s	required steps: 5 time: 8s
2	max distance: 2 dot amount: 1 time: 5s	amount of taps: 1 time: 5s	Shake-amount: 300 time: 8s	required steps: 10 time: 8s
3	max distance: 3 dot amount: 1 time: 5s	amount of taps: 1 time: 5s	Shake-amount: 400 time: 6s	required steps: 10 time: 8s
4	max distance: 4 dot amount: 1 time: 4s	amount of taps: 1 time: 3s	Shake-amount: 500 time: 5s	required steps: 10 time: 5s
5	max distance: 5 dot amount: 1 time: 3s	amount of taps: 1 time: 2s	Shake-amount: 500 time: 3s	required steps: 15 time: 5s
6	max distance: 5 dot amount: 1 time: 2s	amount of taps: 2 time: 3s	Shake-amount: 750 time: 3s	required steps: 20 time: 5s
7	max distance: 6 dot amount: 1 time: 2s	amount of taps: 2 time: 3s	Shake-amount: 1000 time: 5s	required steps: 25 time: 5s
8	max distance: 6 dot amount: 2 time: 4s	amount of taps: 3 time: 4s	Shake-amount: 1250 time: 5s	required steps: 25 time: 5s
9	max distance: 7 dot amount: 2 time: 3s	amount of taps: 5 time: 5s	Shake-amount: 1500 time: 5s	required steps: 25 time: 5s
10	max distance: 7 dot amount: 3 time: 3s	amount of taps: 5 time: 3s	Shake-amount: 2500 time: 3s	required steps: 30 time: 4s

Table 3.2: Table showing how the Adaptability variable influences MinigameSerie2.
max distance: how far the dot will spawn away from the cursor.
dot amount: how many dots you have to hit before completing the minigame.
shake amount: how much the cube has to be shaken before you complete the minigame.
amount of taps: how many times you have to tap the flashing side of the cube.
required steps: how many steps the cube has to detect from your rotating movement.
time: how long you have to complete the minigame.

AV Level	Roadrunner	Flip	Snake
1	Starting-speed: 1s/step speed increase: every 3 steps Change direction chance: 5%	flip-speed: 0.15s required flips: 1 time: 5s	starting-speed: 1s/step starting TailLength: 0 required apples: 1
2	Starting-speed: 0.8s/step speed increase: every 3 steps Change direction chance: 10%	flip-speed: 0.15s required flips: 1 time: 5s	starting-speed: 1s/step starting TailLength: 1 required apples: 2
3	Starting-speed: 0.8s/step speed increase: every 2 steps Change direction chance: 10%	flip-speed: 0.10s required flips: 1 time: 5s	starting-speed: 0.8s/step starting TailLength: 1 required apples: 2
4	Starting-speed: 0.8s/step speed increase: every 2 steps Change direction chance: 10%	flip-speed: 0.10s required flips: 1 time: 3s	starting-speed: 0.8s/step starting TailLength: 2 required apples: 3
5	Starting-speed: 0.8s/step speed increase: every step Change direction chance: 25%	flip-speed: 0.075s required flips: 1 time: 3s	starting-speed: 0.8s/step starting TailLength: 2 required apples: 3
6	Starting-speed: 0.6s/step speed increase: every step Change direction chance: 25%	flip-speed: 0.075s required flips: 2 time: 5s	starting-speed: 0.6s/step starting TailLength: 2 required apples: 3
7	Starting-speed: 0.6s/step speed increase: every step Change direction chance: 33%	flip-speed: 0.075s required flips: 2 time: 5s	starting-speed: 0.6s/step starting TailLength: 3 required apples: 4
8	Starting-speed: 0.5s/step speed increase: every step Change direction chance: 33%	flip-speed: 0.075s required flips: 2 time: 4s	starting-speed: 0.5s/step starting TailLength: 3 required apples: 4
9	Starting-speed: 0.5s/step speed increase: 2 per step Change direction chance: 50%	flip-speed: 0.075s required flips: 3 time: 4s	starting-speed: 0.5s/step starting TailLength: 5 required apples: 5
10	Starting-speed: 0.5s/step speed increase: 3 per step Change direction chance: 75%	flip-speed: 0.05s required flips: 3 time: 4s	starting-speed: 0.35s/step starting TailLength: 7 required apples: 5

Table 3.3: Table showing how the Adaptability variable influences the new minigames.
starting speed: how fast the marker moves, written in seconds per step.
speed increase: how many steps the marker has to move before its movement speed (in seconds per step) increases.
change direction chance: the chance the marker changes direction during the minigame.
flip-speed: how long you have to flip the FC around to count as a success
required flips: how often you have to flip the FC for the minigame to be completed
starting taillength: How long the snakes tail is when you start this minigame
required apples: how many apples you have to eat in the minigame to complete the minigame
time: how long you have to complete the minigame.

4. Results

This section describes all the most important results gathered from the performed experiments. Appendix B provides a complete overview of all gathered data during the experiment.

4.1 Experts Experiment

For the experiment with the experts, we have gathered three experts. All three experts are teachers at the Hogeschool van Amsterdam (HvA). The first expert has expertise in movement therapy and digital health, also called eHealth. The second expert is a child therapist. The last expert is experienced with using blended technology but is mostly familiar with using this with the elderly. All three experts participated in the first part of the experiment, but only the first and third experts mentioned before could participate in the second part of the experiment. This was due to unforeseen circumstances, which meant the second expert did not have time to perform the second part of the experiment.
The experts made clear that the minigames Find The Dot, Rotate, and RoadRunner were great games to train fine motor skills, for the coordination between both hands and requiring a good range of different motions. The tapping minigame was fine, but it would be better if it required more specific points to tap than just the side. Shake was seen as a bad minigame for it required gross motor skills by moving your entire arm rather than just your hand. The experts also made a number of requests, which include the following:

- Add a motion for Pronation/Supination. This was a fine motor skill motion that was not yet present in the first version of the game. This resulted in adding a new minigame that was called Flip.
- Make a version of RoadRunner, where the user moves the dot by pressing the side, instead of having to follow the RoadRunner. This resulted in the addition of a new minigame called Snake.
- Change Shake so you have to shake in a specific direction. This resulted in the minigame called Special Shake.
- Reduce the mental strain on the user. In the old version of the game the minigames would just pop up and it was up to the user to understand what game was playing and what they were supposed to do. This would require more of a person's pattern recognition skills than their fine motor skills. This feedback resulted in the addition of the announcement statements to all versions of the game.
- Make it possible to track the user's finger over the cube and make the user press specific points on the cube. Due to the limitations of the hardware of the cube, it is not possible to determine where on the cube you press a side, so these requests could not be fulfilled in the newer version.
- Force the user to use a specific hand with some minigames, like finding the dot and tapping. Again, due to the limitations of the FC, this could not be enforced by the FC itself. It would still be possible to implement, but it would create an extra layer of complexity for the users, which was something the experts recommended to reduce. Therefore, it was decided to forgo this feedback in the newer version of the FC.

After implementing most of the feedback from the experts, the new versions of the games did not rate significantly better and the new games did also not rate much higher or lower compared to the other games. Additionally, the experts did state the new games were good implementations of the provided feedback, but required some further fine-tuning, just like the original set of games.

The experts did state that the game was quite challenging, really not that boring, but a little bit frustrating (see also Table 4.1). Lastly the experts stated that they would expect users to play with the FC for at least 6 weeks before you would see some improvement in the fine motor skills.

Flow parameter	Expert 1	Expert 3	Average
Frustration	7	3	5
Boredom	3	2	2.5
Challenge	8	8	8

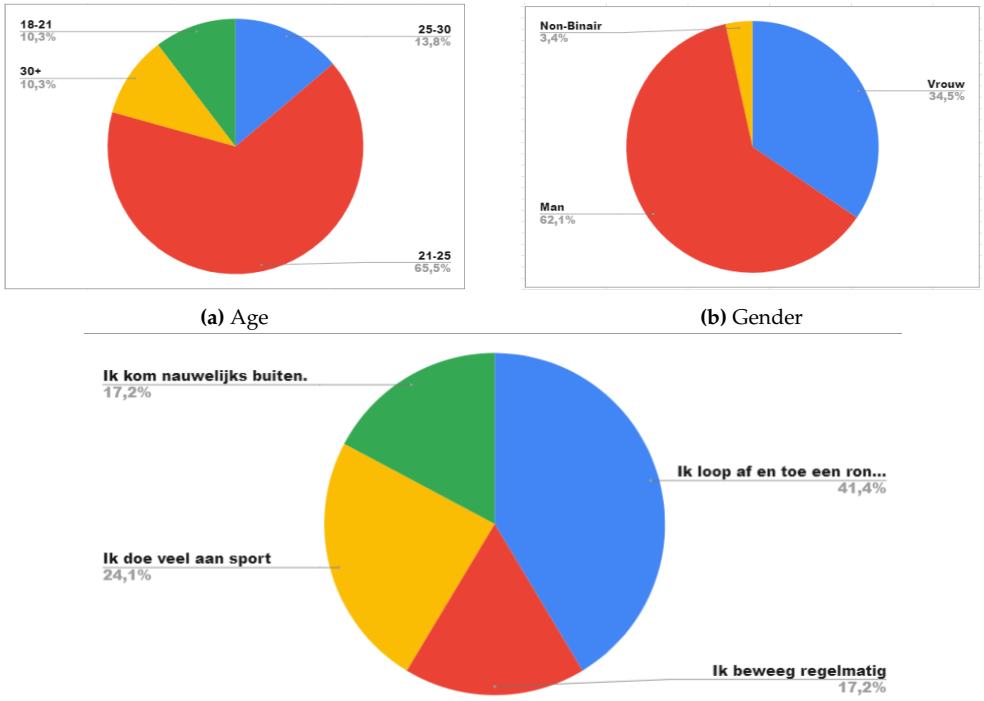
Table 4.1: 10-point scale score in which the experts state how much frustration, boredom, and challenges they experienced while playing the minigames. The questionnaire can be found in Appendix C *Expert 2 is the expert who was unable to perform the second part of this experiment.

4.2 Experiment with the Students/Alumni

A total of 29 students were willing to participate in the second experiment. Most of the participants were 21-25 year old (63.3%) male (62.1%) who performed low amounts of physical exercise (43.3%). Almost none of the participants had physical issues (93.3%) nor had they had family members with a genetic disease that reduces their fine motor skills (90%).

The participants were required to perform some fine motor skills tests before and after playing the game on the FC. As stated before, the participants would have to play with the cube at least for six weeks to see improvement, so these tests are more to see if there are changes in fine motor skills. We are interested to see if there is a significant **difference** between performing the fine motor skill tests before and after playing the game, which means we will perform a two-tailed T-test with the following H_0 and H_A:
H_0: *There is no difference in fine motor skills before and after playing with The Futuro Cube*
H_A: *There is a difference in fine motor skills before and after playing with the Futuro Cube*
Because we test the same people before and after playing with the Futuro Cube, we can perform a Two-tailed two-sampled T-Test. A total of 5 different fine motor tests were performed by the Participants: NinePegHoleTest, Tap, Flip, Knock, and Rotate. These tests are further described in Appendix B. Three of the tests, NinePegHoleTest, Flip Test, and Knock Test, did not result in a p-value > 0.05, meaning that there was no significant change in fine motor skills before and after playing the game. A p-value of <= 0.05 would mean that there is a significant difference from the Null-Hypothesis (H_0), which would mean that the Null-Hypothesis is incorrect and we can assume that our Alternative hypothesis (H_A) can be accepted. Going back to our fine motor skills test, we do however see that two of them did suggest a change in fine motor skill: the Tap Test and Rotate Test both returned a p-value < 0.05. In the case that there indeed should not be any change in fine motor skills, then there are other variables influencing the test results.

Looking at the difficult performance of the games, we have to perform a different kind of test. First off, we are only interested in seeing if the minigames have become significantly more difficult. What's more, we are interested to see if minigameseries3 is more difficult than minigameseries1, which would show that the new additions to the minigames are aspects that make them more difficult. The two groups were split into two almost even groups of 14 and 15 participants, where 14 played with the old version and 15 played with the new version. This results in a one-tailed two two-sampled T-test with:
H_0: *The new DDA System does not make it more difficult than the old version*
H_A: *The new DDA system is more difficult than the old version*
Looking at the results (see also Appendix B), we see that the new Find The Dot game is the only game that is significantly more difficult in the new version than it was in the old version. Of the other games, tap was more difficult, but not significant enough, and Rotate and Shake were even shown to be easier. We do see that the new game version was considered significantly harder.



(a) Age

(b) Gender

(c) Exercise

Figure 4.1: Some Questionnaire results

Both when we just look at only the 4 standard games of find the dot tap, rotate, and shake and if we look at the full game of both versions, we see that the new versions have an increased difficulty with a p-value < 0.05. That being said, if we look at the MinigameSeries3 new version of just the base games of find the dot, tap, shake, and rotate and compare it to the difficulty of the full game, we don't see a significant increase in difficulty. This means that it is not proven that the new games make the game more difficult.

Lastly, we can look at the Flow and Worry of the different series. In these tests, we are interested to see if the participants experienced More Flow or Worry while playing the game than during normal activities. Using the T-Norm table provided by Rheinberg et al[33] (see also Figure B.21 in the appendix) and the cumulative score of the flow questions as described in Appendix C.3.2 we can see if the users experienced a significantly more Flow and Worry while playing the game. With this information, we can perform a one-tailed one-sampled T-Test with:
H_0: *Users don't experience Flow/Worry while playing the game on the Futuro Cube*
H_A: *Users experience Flow/Worry while playing the game on the Futuro Cube*
If we assume that the normal "Flow/Worry" a person experiences is 50, means that any score higher than 50 shows the user experienced some form of Flow or Worry, depending on which questions we analyze. (see also Appendix B for the description of all results and the full analysis). Looking at the Flow and Worry combined we see that at least 68,9% of the participants experienced some form of flow and that 62% experienced worry. However, neither of the scores was significant enough to create a p-value < 0.05. Looking deeper into the two different versions we cannot see a significant enough improvement in flow or worry score to state that participants will experience flow or worry while playing this game, though the scores do suggest that the new version reduces flow and increases worry with the participants.

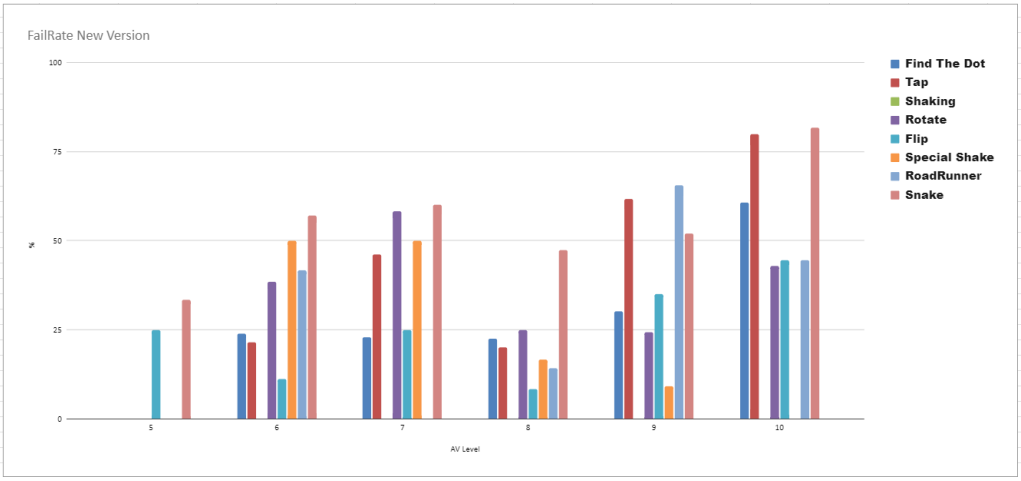


Figure 4.2: % of failure for each minigame over the different AV levels in MinigameSeries3 from the experiment with the students

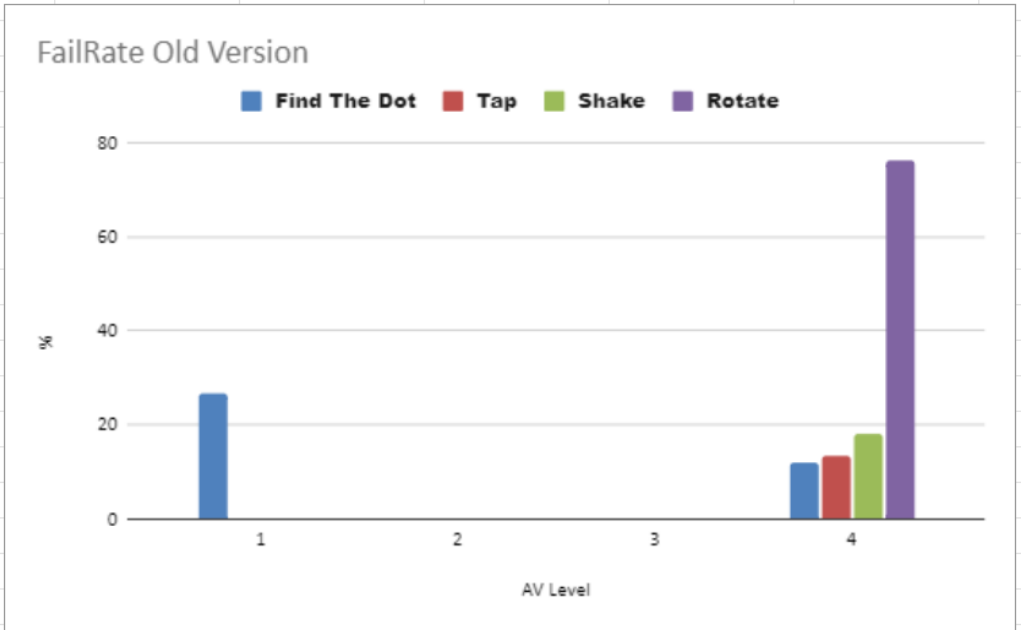


Figure 4.3: % of failure for each minigame over the different AV levels in MinigameSeries1 from the experiment with the students

5. Conclusion

In this section, we will answer our two research questions: "What factors should be adapted in a DDA system to change de difficulty of the minigames?" and "What is the impact of using the Futuro Cube on training the fine motor skills?"

Looking at the results it is clear that one factor has the biggest influence on the difficulty of minigames: Time. The more or less time you have to complete a minigame the easier or more difficult the minigame will become respectively. There are other factors you could add as well. For example, increasing the number of actions the user has to perform before it is considered a success, but this means that you have to scale the time based on the number of actions. If we look at Figure 4.2 we can see that for things like Find the Dot and Tap the fail rate decreases, meaning that the minigames become easier to complete, around the 8th AV Level. It also happens to be that the 8th AV level is the point where the number of sides you need to tap or dots you need to find increases. This shows that the increase in number of actions doesn't influence the difficulty of the games.
More minigames, on the other hand, do make the game more difficult. A bigger range and variety of minigames make it more difficult for users to follow what might be active. However, this comes at the cost of reducing the user's flow experience. Furthermore, it increases the feelings of anxiety among the users while they are playing the game.
So, the required amount of time you have to complete a minigame and the amount of different minigames adapt the difficulty of each separate minigame.

Our second research question we have to leave unanswered, because we don't have the result to support a claim either way. Our experiment time is too short to expect a significant change in fine motor skills with the users, before and after they play with the FC. Our Tap test and rotate Test did show significant enough change in fine motor skills, but this can be explained due to other reasons. The reason the tap test resulted in such a difference in fine motor skills is that the FC is not always as responsive to the users' tap. It took a couple of participants a couple of attempts before they understood how they could tap the FC in a way that would register the required taps or sometimes the second time around the taps would be registered less consistently. This inconsistency was the cause that there was a big difference between the results in fine motor skills. The rotate test was significantly different because of the conscience process bound with the test. The test required participants not to use their wrists while rotating the cube. This not using their wrist would make it so that some participants had more trouble while they consciously tried to avoid using their wrists. Also, the level in what way the wrist was or wasn't used, even with guidance, deviated between the two times the test was performed with each participant. This is the most likely cause of the significant difference detected in the rotate test.
The experts explained that seeing an active improvement in fine motor skills would take at least 6 weeks. These explanations would help us explain why the two tests that did show a significant change in fine motor skills are wrong and why the remaining three tests did not show a significant change, including the Nine Peg Hole Test, which has been proven to be a quick and easy fine motor skill test.
While we cannot say if changing the difficulty could improve the training capabilities of the Futuro Cube, the experts did say that they don't believe it is impossible that it could help, though, in its current state, it will be unlikely to do so. This is due to the limitations of the Futuro Cube that make it not possible to perform more specific motions that the experts would like to see.

In Conclusion, we see that our analysis of the Futuro Cube resulted in showing that the FC has great possibilities for improving a user's fine motor skills. But in its current state it is not good enough for experts to be willing to use the FC in their therapy sessions. For the FC to be considered a great tool to train a user's fine motor skills it needs:

- To be able to train both hands equally and not have one hand as a supporting hand. This can either be done by a system that forces the user to use one hand or make games that use the coordinates of both hands equally.
- To be more precise. This would allow to train a wider range of fine motor skills.
- To have a better DDA system that helps the users enter a better state of Flow and lower state of Worry. This will ensure that the users will be willing to play (enough) with the Futuro Cube without feeling a mandatory task they have to perform.

The second point is hardware and seeing that the FC was created for entertainment purposes and not training purposes it is unsure if a later version of the FC would be made with this in mind. This would either require a collaboration between researchers and the Rubik's Futuro Cube or the researchers creating their whole own FC.
The third point is software-related and can be researched more deeply, this also gets described further in section 6.
The first point is a combination of hardware and software. If the hardware can be improved to detect a specific hand we can use that to train both hands equally. Otherwise, we have to focus on the software side and make better games that use both hands' fine motor skills equally.

6. Discussion

In this section, we go into some more detail about the conclusions we found in the previous section and some possible angles to explore further.

The fact that the number of different minigames and the time available to complete the minigame are the two main factors that influence the difficulty of the minigames suggests that the DDA system made by the old version was doing the correct thing. That being said, the old DDA system had some flows that the newer version did improve upon. To start off, setting a more deterministic time compared to a dynamic time seems to work better. The dynamic time would result that sometimes a minigame is chosen and the user had already failed the minigame before they had time to react to it. By setting some more pre-determined times for the different Adaptability Levels, we can at least make sure that the user has enough time to complete the minigame if they know what they are doing. Another problem with the old DDA system is that it is very unrestricted in changing the AV level. The old DDA system would constantly change between AV levels 1 and 4, skipping the AV levels in between often. A more restricted system that would only allow it to move up and down by one level would make it so that the system would not go from very easy to very difficult very quickly.
That being said, the new DDA system could also be improved upon further. While the minigames are not that complex, it would be possible to see if further changes can be made to make them adapt their difficulty even further. Another thing that could be looked more into is the difficulty of the different AV levels. While most of the participants did state they play video games in their free time, most of the participants playing the new version played for the longest time and ended on AV level 9, the second highest AV level. It is possible that some changes are required to make most of the AV levels more difficult or the different AV levels be more similar so that DDA systems actually adapt to the player's skill level, instead of converging them all to the same difficulty scaling.
Additionally, the old system of filtering minigames could be added to the new DDA system. As said before, the number of minigames influences the difficulty of the separate minigames. If we were to limit the number of minigames based oa the AV level, i.e. what the old DDA system did, we could have a better influence on the minigame's difficulty. This would require another look at the minigames and see what minigames would only be required to appear at higher AV levels.
Speaking about the minigames, Shake should get more attention in a follow-up experiment. It is clear from the experiments that not only is Shake far too easy to complete, but it also does not train the correct physical skills. Shake should therefore be removed from all the series, or should be improved in a way that it requires fine motor skills instead of gross motor skills.
Another thing that could be looked at is making a more complex DDA algorithm. The two used Algorithms 2 and 1 are not the most complex algorithms out there. Working out a better Probabilistic method is definitely still a possibility. As for the earlier proposed Reinforcement learning, it is unlikely that it would be a good direction to go about it. The way the Futuro Cube works is that when you start a game the scripts get loaded in, making it so that the FC does not remember any of its previous runs. With the system loading in the script every time on start and Reinforcement learning requiring too much time to learn what the right difficulty is for the users, it is unlikely to possibly be used in the current versions of the Futuro Cube.

That is not the only limitation that should be looked into pertaining to the Futuro Cube. The experts stated that in its current state, they did not intend to use the Futuro Cube. This was not due to the minigames nor the full game they played on it, they considered them to be decent implementations. The problem with the FC is the inability to more precisely track finger movement. If the FC was able to detect a specific point on the 3x3 matrix instead of just that the side was pressed it would allow for better training of the user's fingers. Also if it were able to track a user's finger over the cube, it could allow new minigames that train a whole new set of fine motor skills motions. If a newer version of the FC would ever be created that would be able to detect these kinds of motions and presses, then some of the Experts stated they would be willing to use the FC in their therapy sessions.

Another thing that could be researched further is looking at what to do with the user's non-dominant hand. Right now, some minigames require only really the use of one hand, making the non-dominant hand often to be a supporting hand. This would mean that the user is only training one of their hands, which is not what we want. There does not seem to be a way in which we could use the FC to detect which hand is being used, so there should be some additional research done to see if there is a way we can encourage users to use a specific hand or researched if it is possible to change the minigames in way that they use both hands equally.

The last and most important thing that should be researched further is to see if the FC improves the fine motor skills at all. While the experts do require some changes to be made on the hardware side of the cube, we still have yet to see statistical proof that the FC can train the user's fine motor skills. The data is promising enough, but we cannot say for certain whether the FC does help the user's fine motor skills at all, without having them play with the FC for at least six weeks, for at least one hour a day. One of the experts suggested how to perform this research: give the participant the Futuro Cube and explain to them how to use it and what they are required to do with it. Then they need to log every day for how long and how well they performed with the game. Then perform some fine motor skill tests and see if there is a significant improvement.

Even if it would come out that the Futuro Cube is not a good enough tool to train fine motor skills, that does not mean that the usage of the FC ends there. Most of my participants do a lot of work behind their laptops, which could lead to RSI problems. The Futuro Cube could be a good tool for these kinds of people during their breaks to maybe reduce the risks of RSI. It could also be a good fidget toy for people, reducing the strain on their wrists and fingers. Whether this is actually the case would, of course, need to be researched more, but it shows another use for the Futuro Cube in case its training purpose comes out as insufficient.

Acknowledgment

Dr. Annette Brons and dr. Michel Oey from the Hogeschool van Amsterdam and Sander Bakker from Utrecht University helped out with this thesis by introducing me to the Futuro Cube and lending me the equipment needed to work with the Futuro Cube. The original idea of this thesis project was also because of dr Annette Brons. Honza Krejsa helped out a great deal by giving me early access to the newest Firmware version of the Futuro Cube (V8.144) and a new version of RFCSuite (V1.9). Honza Krejsa was also the person who helped me out by being able to connect with the Futuro Cube to be able to get data from the Futuro Cube. Pepijn Thijssen helped out by creating and providing the original code for the minigame series used in the thesis. Without any of these contributions, this thesis would not have been possible.

Appendices

A. FuturoCube Minigames

A.1 Futuro Cube Minigames

In this Appendix we will go over the different set of minigames that are present on the Futuro Cube. We also go over in what way the minigames could be adapted to be made more difficult or easier.

A.1.1 Default games

These are the minigames that are by default present on the FC. Short explanations and tutorial videos can be found on the official Futuro Cube website[36]

A.1.1.1 Gomoku

In Gomoku, you try to get five dots placed in a row. You are allowed to place a dot first, after that, the cube is allowed to place a dot. you confirm the placement of your dot by double-pressing the top of the cube. You can move the dot around by rotating and tilting the cube. There is one additional rule where the directly opposite faces are considered connected for diagonal lines.

The way this game can be made easier or more difficult is by changing the rules where the opponent places their dots. For example, a more difficult AI might try to get a diagonal line of 5 because that one is harder to detect or might place its dots in such a way that it blocks the line the player is trying to create. Meanwhile, an easier AI might intentionally not create a line of 5 dots or even intentionally not block you from creating a line of 5.

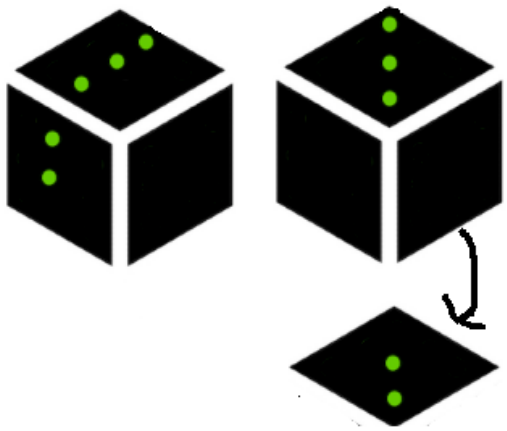


Figure A.1: Two variations of having 5 dots in a row to win a game of Gomoku

A.1.1.2 Multi Gomoku

Same as Gomoku (A.1.1.1), however now the second dot gets placed by a second player with their own FC. Any dot that gets placed by the opponent will also appear on your FC and visa versa. In Section 2 we already talked about how competitive games can create less desirable results, therefore we will not use this game in our study.

A.1.1.3 Road Runner

In Road Runner, a dot on top of the FC will appear and it will start randomly moving around the FC. The goal of the game is to keep up with the dot and make sure that the dot is always at the top side of the FC. You do this by rotating and tilting the FC. The longer you keep up with the dot, the faster it will move (and the more points you score for keeping up with it). If you fail to keep up, the roadrunner will wait and slow down so you can catch up with it. After a set amount of time, a final score gets calculated.

Road Runner could be made more difficult by increasing the chance that the roadrunner will move in a random direction (and easier by reducing that chance), this will require you to quickly change your hand movement and will make it harder to keep up while the dot is running around. Another tactic we could use is decreasing/increasing the speed-up factor, which would allow the users to feel just the right amount of challenge while playing.

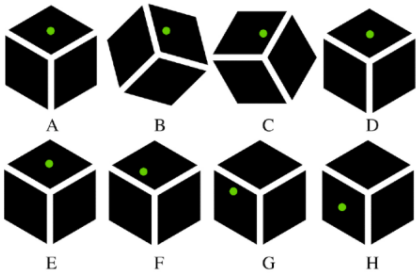


Figure A.2: schematic of the game Roadrunner, made by Brons et al. [2]. Pictures A-D show what the player is supposed to do to keep up with the dot and keep it on the top of the cube and E-H shows what happens if the player fails.

A.1.1.4 Piano

In Piano, you can use the FC music box to play sound. The Piano module allows you to play 6 different sounds based on its orientation. The better you hit the middle of a side the better the sound it produces. Making some good music with this is quite difficult and can require some training. Additionally, because this module is not really a minigame with a clear end goal. Therefore will Piano not be used in this study.

A.1.1.5 Cubris

In Cubris—which is Tetris but adapted to work with the FC— Tetris-like pieces are created on one of the sides of the FC (always the same side, after the game is started). You can press the side of the cube to turn a piece clockwise or counterclockwise, you can tap the top to place a piece. The pieces come in different colors, for example purple, blue, light green, etc. If you fill a face or create a ring around the whole FC, the dots clear up and you get points. If the ring or face is filled with the same color dots, you get bonus points when clearing it up. To pick up a piece you have to rotate the futuro cube in such a way that the side where the pieces spawn is faced towards you. Additionally, the face where the pieces spawn cannot be filled. The game ends when you're unable to place the next piece anywhere on the FC.

Cubris could have difficulty scaling in the following two manners:

- We change the chance certain pieces can spawn so that more difficult-to-place pieces are more/less likely to spawn
- We add a mechanic that you have to place spawned pieces within a certain time limit, and the more difficult we want to make it, the lower the amount of time you have.

Both these changes can result in users needing to rotate the FC more and therefore train the motions more.

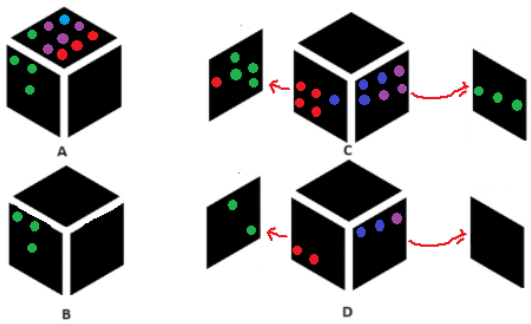


Figure A.3: The two ways you can clear blocks on the cube. A-B shows what happens if you fill a face of the cube and C-D shows what happens if you fill in a line. Different colors are used to show off the different kinds of pieces that can exist

A.1.1.6 Multi Cubris

Same as Cubris (A.1.1.5), however the FC is linked with another FC. Both FCs get the same set of pieces to place in their game and the goal is to clear as many faces and circles within the time limit, without dying. Because of its competitive nature, it is like Multi Gomoku (A.1.1.2) excluded from this study.

A.1.1.7 Gravity Puzzle

In Gravity Puzzle you basically try to solve a Rubik's puzzle. all LEDs will start showing 6 different colors. These colors are in sets of 9, each filling one side. After you tap the cube again it will shuffle the LED colors and you need to try to solve it to get all the faces back to their original way. You can rotate a face clockwise by pressing on it if the face is top-side directed and you can move everything on the face to the left or right by pressing the left or right faces respectively.

This game has no clear features or metrics you can change for difficulty scaling. Additionally, just like with a Rubik's cube, if you're not accustomed to the tactics of solving these kinds of puzzles it can take a long time before you're able to solve this kind of puzzle. For these two reasons, this puzzle is excluded from this study.

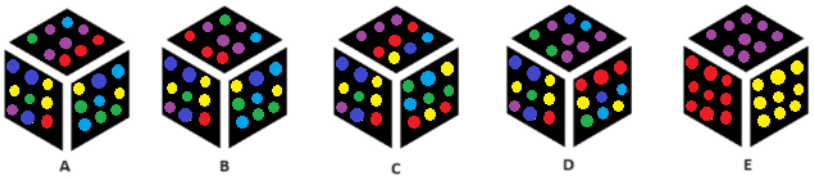


Figure A.4: A shows a random configuration of the puzzle, B, C, and D are the configurations of how the cube will look after you perform one of the actions. B shows you what happens if you press on top of the cube, C what happens if you tap the left side of the cube, and D what happens if you tap the right side of the cube. E is what the puzzle is supposed to look like once you've completed it

A.1.1.8 Ring Puzzle

In Ring Puzzle we try to achieve the same thing as Gravity Puzzle(A.1.1.7): The cube will randomly shuffle 6 different colors in sets of 9. The goal is that each side is filled with one singular color. The way you move the dots around is however different. Each middle point is fixed and cannot be changed. This means that each side has a designated color beforehand. You can rotate the dots in two ways:

- you can move the dots on the face itself
- you can move the dots along a line on the cube

Figure A.5 shows schematics that show these two possible rotations. The way you choose between the two different rotations is by tilting the FC. By tapping the cube after it lights up one of the two options it will rotate the "ring" you have chosen.

Because this is closely similar to Gravity Puzzle it also suffers from the same problems as Gravity Puzzle. It will therefore be excluded from the study.

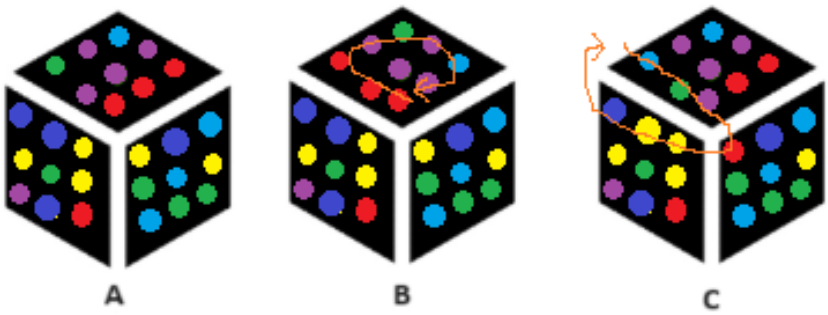


Figure A.5: Two different ways the dots can move around in Ring Puzzle

A.1.1.9 Connect

In Connect, the goal of the game is to connect two center-symmetric sides with one another through a line of dots. Just like Gomoku A.1.1.1, you and the FC (AI) take turns placing dots and your goal is to create a fully connected line between the center of two different faces. Diagonal-placed dots are not considered connected in this game mode.

Just like with Gomoku the way we could adapt this game is by improving the game's AI. If it knows the best moves it can take, it can then decide based on the difficulty if it intentionally makes bad moves or makes the most optimal moves.

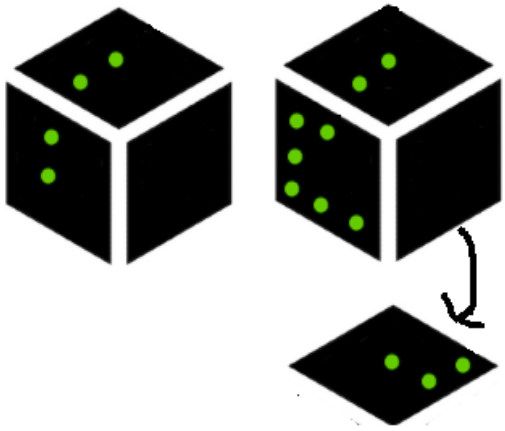


Figure A.6: Two ways where two different center-symmetric sides are connected to one another to win a game of connect

A.1.1.10 Multi Connect

Same as Connect (A.1.1.9), however now the second set of dots gets placed by a player with another FC. Any dot that gets placed by the opponent will also appear on your FC and visa versa. Just like the other multiplayer games, this game is excluded from the study because of its competitive nature.

A.1.1.11 Snake

In Snake, you appear as a series of dots that slowly grow as you eat more apples. There are two kinds of apples: poisonous apples and normal apples. Normal apples are flickering red dots and eating them will let your snake grow and nothing more. Poisonous apples are flickering green dots and when you eat them you'll temporarily lose control of the snake. The snake usually will move towards the top of the Futuro cube, so by tilting and rotating it around, you can move the snake around the cube. However, after you eat a poisonous apple you'll move in a straight line for a while. Still, the apples—including the poisonous ones—only disappear after the snake eats them and new apples only spawn after the one before is eaten. The game ends when the snake ends up hitting himself with his movement.

This game has a couple of options for difficulty scaling

- We can increase/decrease the rate at poisonous apples spawn. Losing control of the snake can be very dangerous, making you to plan ahead when you need to eat these apples, otherwise, you can end up running into yourself
- Change the apple spawn rates. To make it more difficult we might spawn the apples close to the snake its body so he has the risk of hitting his own body. On the other hand, we might place it in a more open area if we want to make it easier for the player.
- Increase the snake's movement speed. Either when the player chooses to begin the game or while playing the minigame (for example, after every apple he eats). By increasing the speed the snake is moving around, we force the player to react quicker to make sure the snake doesn't hit himself.

Each of these options needs to be tested to see what of an effect they would have on the playing style of the player.

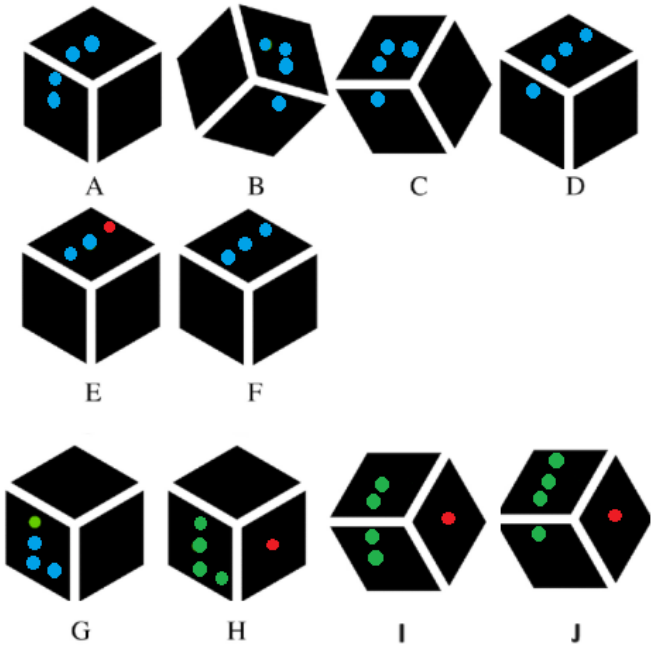


Figure A.7: A-D show how the snake moves in the game Snake. B and C show how you should tilt the cube if you want to move it to the left or right. D shows how the snake will move if you don't tilt the cube. E-F shows what happens if the snake eats a normal (red) apple. G-J shows what happens if the snake eats a poisonous apple, the snake will continue moving upwards, even if the player rotates the cube.

A.1.1.12 Gravity Challenge

In Gravity challenge. You have 3 different colors in sets of 11 and your goal is to connect all the same colors with one another. dots are also considered connected if they are attached to each other over different faces that are not diagonally linked. Because we only use 33 of the 54 dots, we have a couple of empty spaces. Each time you tap the cube all the dots that can "fall down" move one spot down based on which way the cube is rotated. This can make the dots move to another face, if possible.

This puzzle however suffers from the same issues as Gravity Puzzle (A.1.1.7). It will therefore be excluded from the study.

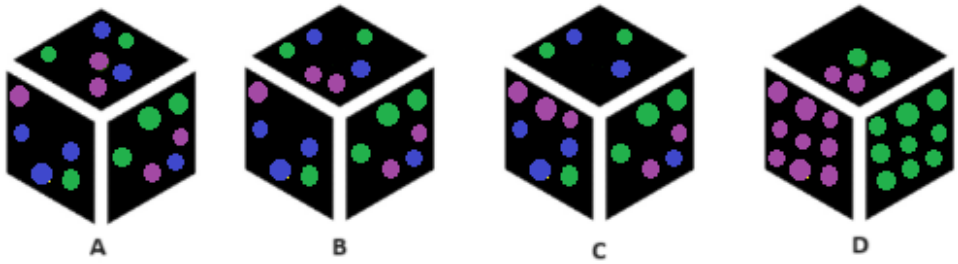


Figure A.8: A shows the starting position of the dots, B shows what happens if the dots go down one place on the face itself and C shows what happens (from B) if you let the dots fall from one side to the next. D shows a possible end result you would want to achieve in this game.

A.1.1.13 Dream Quest

In Dream Quest, the Cube generates a random invisible maze and you as the player need to walk through it by tilting the cube. If you hit a wall, the cube will make a sound and you stay on the same part of the maze you were before. Your goal is to find different sound spots that are present around the maze. Each of these sound spots plays a note on a different volume. Your goal is to find the spots in order from lowest to highest volume.

For difficulty adjustment, we have a couple of options

- increase/decrease the size of the randomly generated mazes
- place the sound spots closer/further from one another
- increasing/decreasing the amount of sound spots the player has to find

These changes are mostly things that one can do before the mazes are made, not while a player is playing. There is an option however to help the player while playing. We could change the position of the sound spots if the player is struggling to a closer unexplored spot if this music spot hasn't yet been found. We can even go as far as spawning the music spot on the next unexplored space. We could also do the opposite and force a player to find x amount of spots in the maze before he finds a spot where the next music spot is. However, this game requires more brain power than using the FC for its motor skill training, therefore we will just exclude it from this study.

A.1.1.14 Sokoban

In Sokoban, your goal is to push boxes to a certain goal location. You as the player appear as a blue dot, the boxes appear as red dots, the goal location of the boxes appears as green dots and the walls of the puzzle appear as Purple dots. If you push a box to its goal position it will light up light blue. The face the player is present on is considered the top face and pressing one of the sides allows the player to take one step towards the face that was tapped. If a box (red dot) is in front of the player then it will be pushed in the same direction. The player and box can be pushed to other faces this way, but they cannot move or be moved through a wall. If the player makes a mistake, it can press the bottom face and it'll undo the last move the player performed. The player wins if they manage to change all the green lights to light blue.

This game suffers from not being able to do great difficulty adjustment at run time, but it is possible to make some preset puzzles, each having its own set difficulty level. This way you could give a player a puzzle fit for their own difficulty level. That being said, like Gravity Puzzle, this game suffers from the fact that, depending on how well-known you are with these kinds of puzzles, it can take it long time to complete them. This game also suffers from that it doesn't require many fine motor skills. Therefore this game is also excluded from this study.

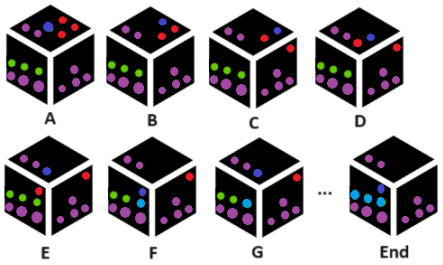


Figure A.9: A game of Sokoban on the Futuro Cube. A-G shows a series of moves the player can perform to move the player and push boxes toward the green dots, with the goal of moving all three (red) boxes to the (green) goal points.

A.1.2 Downloadable games

These are the games that are not directly present on the Futuro Cube but are published games by the community you can download on your Futuro Cube through the official Futuro Cube website.

A.1.2.1 Minesweeper

In Minesweeper you try to uncover all the fields without a mine, while not trying to uncover any field with a mine. You have a white dot which is your cursor, you can move the cursor around by tilting and rotating the cube. If you press the top of the cube it will mark the field with a flag. If you press any other side of the cube it will uncover the highlighted field and depending on its surroundings will give the following colors:

- black, no mines adjacent
- green, 1 mine in one of the adjacent 8 fields surrounding the uncovered field
- blue, 2 mines adjacent to the uncovered field
- yellow, 3 mines adjacent
- magenta, 4 mines adjacent
- purple, 5 mines adjacent
- skin orange, 6 mines adjacent
- lily, 7 mines adjacent
- sky blue, all adjacent fields of the uncovered field are mines.

Like with other games, this game is difficult to perform difficulty adjustment during the game, but you can set a difficulty ahead of playing it. This game requires more thinking power than actually using much hand movement, which means they don't train their fine motor skills much. For this reason, this game is excluded from the study.

A.1.2.2 Kenneth's RELAX

A module that makes it possible for the cube to play soothing music while it slowly changes color to make you relax. This module has no use outside of this purpose, so we can exclude it from our study.

A.1.2.3 PaintMyCube

A module that allows you to personalize your cube. This module suffers from the same issues as Piano with that this module is more to have a bit of fun with the cube and does not really have a set end goal. Therefore it is not important for what we are researching and we can exclude it from this study.

A.1.2.4 Space Alert

Allows you to use the FC in the board game Space Alert. However, because this study is about using the FC directly and not as an addition to a board game, we have no use of this module and can exclude it from this study.

A.1.2.5 Multiplayer Timer

This module will slowly count down and allow you to have multiple different timers running at the same time. This module can be used like a clock in a chess tournament, to slowly count down until one of the timers reaches zero. This module is supposed to be used while something else is taking place, which is not what we try in this research. So this module is excluded from this study.

A.1.2.6 Hundred Sided Dice

Shake the cube to make it roll a one-hundred-sided die. This module does nothing else, so it can be excluded from this study.

A.1.2.7 Multi Purpose Dice

allows you to roll a d4 (dice with 4 sides), d6, d8, d10, d12 or d20, by shaking it. This module does nothing else, so it can be excluded from this study.

A.1.2.8 Lights Out Cube

In Lights Out Cube, the Cube will randomly have red and blue lights active. The goal of the game is to make all the lights on the cube the same color. You can do this by pressing one of the lights. This will switch the color of the light you pressed and the 4 (non-diagonal) adjacent lights to the opposite color—i.e. blue to red and red to blue.

This game can suffer from the same problems as Gravity Puzzle (A.1.1.7) with that you need some tricks to more easily get to the solution. With that being said, unlike the Gravity puzzle, this puzzle is usually easier by just getting lucky and eventually switching to the correct color. Additionally, like some other games described earlier, this game might not have good ways of adjusting its difficulty during gameplay, but it's possible to set the difficulty beforehand. Still, because of these reasons it is not included in this study

A.1.2.9 Animated Rubik's

Allows you to set up a real Rubik's cube simulator. You're able to tapping rotating the faces like you would with a real rubiks cube. This game suffers from the same problems as described in Gravity Puzzle A.1.1.7. Therefore we excluded from the study.

A.1.3 Added games

This section goes in more detail what games have been added by Pepijn Thijssens in the previous research done with the FC and what games have been added by me for this thesis.

A.1.3.1 Pepijns minigames

Pepijn Thijssens was an University Utrecht student who had worked with the Futuro Cube before and had made his own set of small minigames that the player needed to complete as fast as possible. The game itself is basically an amalgamation of 4 different minigames:

- Find the Dot: where you have to move a dot to a specific point on the cube by rotating and tilting it around.
- Tap Side: One of the sides will start flashing and the player has to quickly tap that side with his finger.
- Spinning: A red spinning circle starts rotating around the cube and you have to rotate the cube a couple of times in the same direction.
- Shaking: the whole cube starts flashing and you have to shake the cube in time.

These minigames quickly rotate each other and you have to complete a total of 54 of them successfully to complete the game. For each successful minigame, one of the dots of the Futuro Cube will light up green. A video of this game can be found in [37].

This game has already DDA added to it. It has two mechanics to change the difficulty of the minigames:

- The time you have to complete each minigame is scaled based on how well you perform. This means that if you fail to complete it in time, the time you'll have for the next minigame will be increased. Similarly, if you manage to complete the minigame with a lot of time over, you will need to complete the next minigame faster. Of course, not every minigame takes as long, so there is some additional time balancing added to make sure that it is possible to complete a minigame within the amount of time given.
- Some of these minigames are considered more difficult to complete than others. For example, Spinning compared to Shaking is a lot more difficult to complete. So if the player is performing badly, some of the more difficult minigames get (temporarily) excluded from being picked by the random minigame-choosing system. When the player performs well, then all minigames have a chance of being chosen.

We could however work further on the already-built DDA mechanics, some examples we could implement are:

- Tap Side: instead of tapping one of the sides you have to tap multiple different sides that light up in quick succession after you tap one of the sides
- Spinning: depending on how well you do, you have to make more or fewer rotations to complete the minigame.
- Shaking: you have to shake for longer/shorter or you have to shake more violently/less violently to shake enough in the required amount of time.
- Finding the dot: now, the dots appear on places that are not yet lit up as a signal of completed minigames, we could instead change it so that the dots can appear anywhere. Then, if you play well it will appear far away, while if you play poorly it will appear close by.

A.1.3.2 Maze

A randomly easy or hard maze gets created and the goal of the player is that to move the dot from its starting location to the other side of the cube, without hitting a wall.

This game on its own does not have many options for changing difficulty apart from creating the maze at the start. We can however consider using a mechanic from Dream Quest (A.1.1.13). We will auto-generate a maze while the player is playing the game. The goal of the game is to find the exit of the maze in as fast enough time as possible while penalizing him for hitting walls. Depending on how the player is playing we can make the maze more difficult or easier for them. We can force the player to make more turns inside the maze if he is playing well and we might generate an exit sooner if the player is struggling.

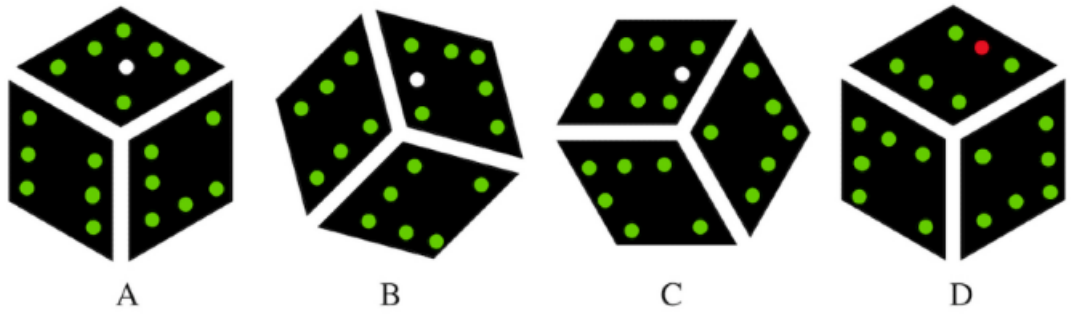


Figure A.10: schematic of the game Maze, made by Brons et al.

A.1.4 Conclusion

After this overview we have a good idea of set games. We can separate the games into 3 categories:

- Games we can perform DDA to possibly have a positive effect on training fine motor skills
- Games we can add DDA, but the games are more logical based, so they won't train the child's fine motor skills much.
- Games that are excluded from the study.

For the first category, games we can adapt and improve upon we have a small list: Snake, Roadrunner, Maze, and Pepijns minigames. Roadrunner and Pepijns minigames can be easily explained, we have some good possibilities to make the games difficult during run time and they make use of making lots of motions with the cube, so it's a good use for training fine motor skills. Snake also makes use of motor skills and has some possibilities to make the user more actively use their fine motor skills. Maze is probably the strangest addition. Using Maze like it was originally created would not work for our purpose, but if we add the adaptation that the maze is continuously generated while playing and that the goal is to reach the end as fast as possible, we can install metrics to change the difficulty at runtime and use the motion controls to push the user to train their fine motor skills.

In the second category, we have the following games: Gomoku, Cubris, Connect, and Lights Out Cube. Each of these games has a possibility to add DDA to them. However, if we think back to the original goal of this thesis, we should rethink whether it would be worth implementing it. We want the users to train their fine motor skills, however, these games require less of the user to actively move the cube around and more to think logically about it. For this reason, we will focus more on the games of the first category and see if there is something of these games we can use when making new games or improving the games.

The third category includes all the remaining games that have been mentioned in this appendix. Each has already been explained why they are not fit for this study.

B. Full Result Enquête

This appendix goes over all the results we have gotten from the different experiments that have been performed. The two experiments were a 2-part experiment with the experts and an experiment with students/alumni.

B.1 Experiment results with the Experts

The Experiment with the experts was split into two parts. After making the first version of the game on the FC, the experts could try out the game. Afterward, they would tell what they liked about the current version of the game and what they would like to see improved. Based on this feedback a new version of the game would be made and we would return to the experts. These experts would then play the newly improved game and they would tell what they considered an improvement and what could be improved even further. This feedback would then be incorporated into the second experiment with the students and alumni.
In total 3 experts were willing to participate in this experiment, but one of the experts was, due to external factors, unable to make time for the second part of the experiment. They were however asked some additional questions to get some extra insightful data.

B.1.1 Expert Experiment, Part I

The experts would get a chance to play each of the different minigames separately and give some direct feedback on each minigame. After they had played all the different minigames separately, they were given the game in which these minigames would be shuffled and played in a random permutation.
After they completed the game they could look over their answers again and give some final feedback on the first version of the game. They were also asked if they would like to see some new games training or using a specific motion not yet present in the game. They were also asked if there were some variables or parameters that should be collected when doing the test with the students/alumni.

B.1.1.1 Minigames Feedback

For the full Questionnaire used in the first part of the experiment see appendix C. Based on the average scores on how useful the minigames were for training fine motor skills based on the expert's opinions, we see that Rotate scored by far the best among the 5 played minigames. According to the experts Rotate requires most of the user's fine motor skills for it requires you to make a lot of movements with both your hands and it requires good use of your fingers and wrists. The only downside of the minigame is that the visualization makes it not sometimes hard to follow which way they are going and wants you to rotate the cube.
On the second place, we have the Find The Dot minigame, though that is mostly due to one expert giving it a low score because a sticker on the Futuro Cube—which was later removed—confused them during the minigame. This minigame scored overall very well because how the dot moves over the cube forces the user to use much of their fine motor skills. One stated downside by the expert was that the movement of the player is counter-intuitive to what you would expect. One other issue was with the visualization of the game. During the game (and minigame tutorial) parts of the cube would light up green to dictate a success, where the goal would be to fully light up the cube with green dots. However, some experts thought you were not allowed to move over the green dots and got very nervous towards the end of the game because they thought they did it wrong by moving over the green dots.

The newly added RoadRunner managed to end up with a score of 3 out of 5 on the usefulness, though this is harder to explain why. One of the experts scored it a 2 out of 5, but did not state what the reason was for this score. Furthermore, they said they found it a very good minigame for training fine motor skills and even went on to suggest making no changes to it in a later version. The other experts stated that they liked RoadRunner mostly for the same reason as rotating: that it requires a good amount of use of both fingers and wrist interaction. One of the experts also stated that they liked how the RoadRunner would start moving faster the better you perform, making it feel like the game adjusts to the players skill level. Another Expert suggested adding functionality where the RoadRunner would not move on its own, but based on the user pressing the sides of the cube and that the goal would be to bring the RoadRunner to a certain point on the cube. This feedback led to the creation of the Snake in the newer version.
Tapping was considered not that great by most of the experts. The biggest described problem with tapping was that it seemed more like a reaction test than a tool to train using your fingers for the tapping action. Another problem was that the FC did not always register your tap, even if you did do it correctly, which could lead to frustration. Lastly, the problem was that the tapping would only require one hand, which would mean that you would often tap with your dominant hand and use your non-dominant hand only as support.
Shaking was by far considered the worst minigame of the set. All the experts agreed that the shaking motion could not be considered fine motor skill, and more gross motor skill. One of the experts did state that one use for it could be to train the user's grip strength, but that this would be of more use to people with disabilities than children and that it would be more useful to do this in combination with another task. A last comment that an expert suggested was to make some variations on the shaking, where you would have to shake it in a specific direction. This lead to the creation of Special Shake Minigame.
An overview of the raw scores can be found in Table B.1.

Minigame name	Expert 1 Score	Expert 2 Score	Expert 3 Score	average score
Find the Dot	4	2	4	3.33
Tapping	3	2	3	2.67
Shake	2	1	3	2
Rotate	3	4	5	4
RoadRunner	4	3	2	3

Table B.1: Average score (with a 5-point scale score) on how helpful the different minigames are for training fine motor skills

Of the 5 Minigames, RoadRunner was the only game all the experts agreed upon that should not be adapted in the next version of the game. As for minigames that the experts would like to see changed there was less consensus. However, two minigames, Find the Dot and Shaking got the most recommendations for changes. See Table B.2

Expert	Recommended to improve	Recommended to not change
Expert 1	Find The Dot, Shaking and Rotating	RoadRunner
Expert 2	Tapping and Shaking	Find The Dot, Rotating, RoadRunner
Expert 3	Find The Dot	RoadRunner

Table B.2: Each Experts opinion whether minigames should be improved on stay the same in the next version

B.1.1.2 Recommendation on New Motions

As for the final question whether the Experts had some recommendations for new motions. One recurring motion they suggested was Pronation/Supination, the rotation of your entire arm. This resulted in the newly created game Flip in the final version.
Further recommendations were a broader use of the user's fingers, for example requiring a user to use a specific finger or user to hit a specific point on the cube (instead of hitting a specific side), or to make the cube trace the user's finger based on where it is pressed. However, due to the limitations of the cube, we are unable to detect these kinds of specific inputs. The FC can only detect if the side is pressed, not where it is pressed. Additionally, it would be impossible to know which finger is used, so this could not be enforced by the software itself and can only be asked of the user directly. This would be considered to fall outside of the scope of this research so these changes have been left out.
A last recommendation had to do with Tapping and Shaking minigames. Because these minigames only require one hand, you would end up using one hand as a supporting hand, thereby only training one hand. The experts wondered if it was possible to force the user to use a specific hand to perform these actions. Enforcing would not be possible, because of the same limitations as described by the last point: the hardware has no way to detect which hand would perform the action. Still, A version was made in which these minigames would request a specific hand because using one hand for some minigames could be a great way to train the user's fine motor skills better. However, after having made this version of the game it became very early on clear that this would cause a lot more thinking power to keep up with the game. Considering that the goal of this game was to train the user's fine motor skills and not to train the user's reaction time on knowing what to do, we forgo the new version and accepted that it would mean that one hand would be used more than the other.

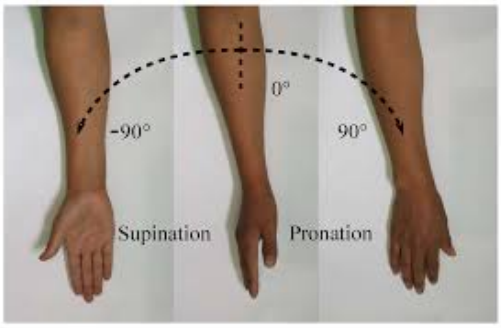


Figure B.1: Arm showing the pronation and supination motion. Rotating your arm in a way your palm is directed upwards is called Supination where as rotating your arm to have the back of your hand directed upwards is called Pronation.

B.1.1.3 Recommendations on gathered data

As for any additional data that should be gathered in the follow-up experiment with the students and alumni, the experts recommended that the most important information would be to gather the time the user takes to complete the different minigames. One expert also recommended gathering some background information (age, hobbies, if they suffer from some physical problems, etc.) to see if that would have some influence on the results.

B.1.1.4 Additional Recommendations

One last recommendation given by one of the experts was to add some announcement module to the game. It would not always be clear which minigame is playing, which would mean you need some time to register what you need to do before you can do it. This would require more reaction time/puzzle solving, which was not desirable. They recommended making it more clear which minigame was up, so the focus could remain on training fine motor skills than learning how the different minigames look like and the user's reaction time.

B.1.2 Expert Experiment, Part II

In the second part of the experiment, the experts would start by first trying out the new games that were made based on their feedback. These minigames were Flip, Special Shake and Snake. Afterward, they were asked to play out the new version of the game where all 8 games get shuffled and played in a random permutation. Once they played through the new game version they would be asked first with which games they noticed a change and again with a 5-point scale score. Afterward, on a 10-point scale score, they were asked how frustrated, bored, and challenged they felt playing the new game. Following this, they could again give some feedback on what they considered improvements and in what way they would like to see even more improvements. Lastly, they were asked what an appropriate time frame would be to see some changes in fine motor skills if they would use the FC for a longer time. This full extend of this questionnaire can be found in Appendix C

B.1.2.1 Minigames feedback

The two new games, Flip and Snake did not score extremely better or worse compared to the original 5 games. Flip scored an average of 3 out of 5 and Snake scored a 3.5 out of 5. Both were described as being good implementations of the feedback that was given, but there was also still some room for improvement with both. This had mostly to do with the recognition of the two systems, both Flip and Snake had the intention to not react to the user's input the way they would have liked.
As for the new versions of the older 5 games, each scored slightly better than before. However because not all had active changes made to them, this can be difficult to explain why they scored better. Because of the very small sample size (2/3 experts), good results can not be derived from it. Furthermore, the increase in score could be explained by the absence of one of the experts.
Out of the 10-point scale score, we got an average of 5 on frustration, 2.5 on boredom, and 8 on challenging. Overall this is a good score, it shows that the DDA system the experts tried managed to adapt well to make it challenging, but not impossible and not too easy. That being said, the sample size is again far too small to get any sensible conclusions out of it.
see also Tables B.3 and B.4 for full list of the new scores.

Minigame	Expert 1 Score	Expert 3 Score	average Score
Flip	3	3	3
Snake	4	3	3.5
Find the dot	4	4	4
Tapping	3	3	3
Special Shake	3	2	2.5
Rotate	4	4	4
RoadRunner	4	4	4

Table B.3: 5-point scale score of the new games and the improved games. For shake the new shake version was used, in which you have to only shake in a specific direction. *Expert 2 is the expert who was unable to perform the second part of this experiment.

Flow parameter	Expert 1	Expert 3	Average
Frustration	7	3	5
Boredom	3	2	2.5
Challenge	8	8	8

Table B.4: 10-point scale score to detect frustration, boredom, and challenge of the experts *Expert 2 is the expert who was unable to perform the second part of this experiment.

B.1.2.2 Additional Feedback

The newly added sound module was a great success with the experts. All asked experts agreed that the custom sounds that announce what minigame is playing are a great help in understanding what they need to do. One of the experts did say that the FC in its current state would not be useful in therapy. If a later version would get improved hardware, allowing it to detect more precise movements and taps, it could have great potential, but now it's still a bit lacking.

B.1.2.3 Long term usage

All three of the experts were also asked how long they would think someone would need to use the FC before a change in fine motor skill would be noticeable. They all agreed that at least four weeks, using it at least 1 hour a day would be needed before you would have a noticeable difference. Considering the limited time for the experiment, this could not be prepared as a follow-up experiment. However, one expert did give one good description of how this experiment would look like: you would hand out the Futuro Cube with the game to the participant. You explain how the cube works and what is requested from them. They have to keep a log of which days they play with the Futuro Cube and for how long. They would also be asked to keep a log of their score to see if there is some improvement to see in their score. If you then perform a fine motor skill test before and after this experiment period, you should be able to detect if the game would influence the user's fine motor skills

B.1.3 Difference Scores Part I and II

Table B.5 shows the different scores the two experts gave to the two different versions of the 5 minigames that were present in both versions of the game on the FC. In this table, you can see that the score for most minigames stays stable so the reason the average score of the minigames is higher is more likely caused due to the missing values of Expert 2 than to better-scoring minigames.

Minigame	Expert 1 OV	Expert 1 NV	Expert 3 OV	Expert 3 NV
Find the Dot	4	4	4	4
Tapping	3	3	3	3
Shake	2	3	3	2
Rotate	3	4	5	4
RoadRunner	4	4	2	4

Table B.5: Table showing the difference in score for each minigame between the two versions. OV stands for Old Version and NV stands for New Version *Expert 2 is the expert who was unable to perform the second part of this experiment.

B.2 Experiment results with the Students and Alumni

The target group of this experiment was 21-30 year old students who are currently in their bachelor or master, or have recently finished their studies. After asking around in multiple places if participants were interested in performing this experiment, a total of 29 participants were willing to participate in the experiment. The experiments were spread out over 2 whole weeks and each experiment would take about 45-60 minutes. 14 participants were given the original version of the game (minigameserie1), with only one modification added to it: the custom sound module. The other 15 participants were given the new version of the game (minigameserie3), which included the new version of the minigames also present in the old version, the newly added minigames and also the custom sound module. Before the participants would be asked to play the game, they would first play some small tutorials teaching them how each of the different minigames worked. After this tutorial they would play the game twice. The first time was to try it out and get some guidance if they needed it. The second time where the player had to complete the game on their own, while their data was collected.

Before the participants would play the game, a couple of small tests would be performed to detect the participants' fine motor skills. These tests included the Nine Peg Hole Test (NPHT) on a game board and a tap, flip, knock, and rotate test on the FC. These small tests would be performed once before and once after the participant played the game. The goal of these tests was not to detect improvement in the users' fine motor skills. The experiment time we had with them would be far too little for that, as also described in section B.1.2.3. The goal of these tests was to see if there would be a significant difference between the tests before and after playing the game. If this would be the case, then it is more likely that another reason would explain seeing improvements in fine motor skills.
Throughout the experiment, a questionnaire would also be given to the participants to fill in. The first part was given at the start of the experiment, in which the participant would be asked some personal questions to see if environmental factors could be of influence on the test results. The second part of the questionnaire would be asked after the users played the game. This part of the questionnaire exists out of 13 statements in which the participant had to indicate on a 7-point scale how much they agreed or disagreed with it. This part of the questionnaire was based on Rheinberg et al. Flow Short Scale [33]. This questionnaire would be used to help answer the second research question (*What factors should be adapted in a DDA system to change de difficulty of the minigames?*) to see if the proposed solution would work well.
An overview of the whole questionnaire can be found in Appendix C.3. All data was anonymized through a 4-digit, randomly generated experiment number, which also the researcher cannot directly translate to a specific participant. Each experiment number had also the addition of the letter at the end: O and N, to differentiate between the new game version (N) and old game version (O).

B.2.1 Version differences

The new game version (i.e. minigameserie3) is an adaptation based on the old game version (i.e. minigameserie1).
The old game version is an almost unaltered version of *Pepijns minigames*, with only two minor changes. The first change was the addition of some debug print statements used throughout the experiment that were not visible to the participant. These changes were only done for the researcher to retrieve valuable information throughout the experiment. The other change was the improved sound system. In section B.1.1.4 it was stated by an expert that adding some Announcement sounds would be of great aid to understanding what you would need to do in the games. This was added in the new version of the game, but eventually, after some consideration was also added to the old version of the game. When the experts were playing with the Futuro Cube, it was clear that even with explanation it would sometimes be hard to follow or remember what you would need to do if a minigame came up. The sound module made it far clearer for the experts what was expected of them, which is ideal. However, this addition of the sound module has nothing to do with the DDA system, which was the goal behind the experiment. By adding this sound module also to the old version of the game we remove a variable that could have some influence on the results while not being linked to the actual DDA systems we're trying to test.
The old version consists of 4 games: Find the Dot, Tapping, Shaking, and Rotating. The explanation of these games can be found in Appendix A.1.3.1. Nothing has been changed on this in the old version of the game.
In the new version, there are already some small changes made to each of the games with how they work, but most are bound to the new Adaptation system described in section 3.5.9. There is one additional change made to the new version of rotate that is not described in that section. In the old version, rotate exists out of 3 lines of dots that move in circles around the cube. The problem with this was that if the 3 dots were not perfectly aligned with the middle of the cube, that one line would rotate only on one side of the cube. If you tried to mimic this specific rotation, then the system had trouble detecting the rotating motion. In the new version of the game, the rotation was reduced to be a singular line only, which allowed the users to better see which way you should rotate the cube. The new version also added the 4 new games of RoadRunner, Flip, Special Shake, and Snake, which is also described in section 3.5.9

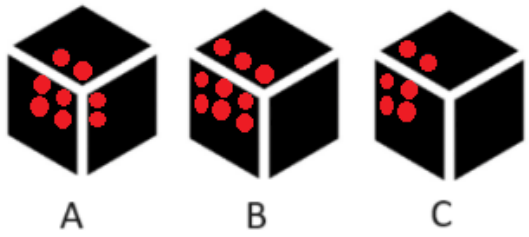


Figure B.2: If you would see the dots like shown here that move in the order A–>B–>C, then trying to follow the dots from the top of the cube would not make the correct rotation movement to detect it.

The last difference was the difference in the DDA systems. The old version had two simple DDA systems built in. The first system was a time-scaler. The better you performed, the faster you had to complete the next minigame. While there was a minimum threshold on how low the time could become, it would get eventually so fast that it would be impossible to complete the minigame. A second built-in DDA system was some form of difficulty scaling. The minigames were considered to be different in difficulty. If you would go from easiest to most difficult minigame you would have: Find the Dot, Tapping, Shaking, Rotating. The old system would randomly decide a minigame, but if your Adaptability Variable (AV) level was low, it would prevent the more difficult minigames from being played and instead would play Find The Dot. Pseudocode of this algorithm is shown in Algorithm 2

Algorithm 2 Adapt Minigame Algorithm. The minigames go from a 0-index with:
0: FindTheDot
1: Tapping
2: Rotating
3: Shaking

```
1:  procedure UPDATEADAPTABILITY
2:      AV ← current AV level
3:      Minigame ← GetRnd(5)
4:      if Minigame == 0 then
5:          <do nothing>
6:      if Minigame == 1 and AV <= 1 then
7:          Minigame = 0
8:      if Minigame == 2 and AV <= 3 then
9:          Minigame = 0
10:     if Minigame == 3 and AV <= 2 then
11:         Minigame = 0
```

The DDA system in the new game version works very differently. Instead of scaling the time, we will now have a set amount of time for each minigame. This time will increase or decrease to make it easier or harder to complete the minigame. Changing this time-scaling method to be more deterministic than variable means that the minigames have at least always a chance to be completed, which is not always the case in the old version. The second part of the DDA system is that, instead of giving you harder or easier minigames, the new DDA system will make the current minigame harder or easier. A full description of those changes can be found in section 3.5.9.

B.2.2 Fine Motor Skill Tests

We have performed 5 small fine motor skills tests with all participants, once before and once after they have played the game.

B.2.2.1 Nine Peg Hole Test

The Nine Peg Hole Test (NPHT) is a test to measure finger dexterity by the participant. This test is mostly used for different neurological diagnoses but has also been used before to test fine motor skills with children [38]. For the test we used a wooden board with nine small holes in it and 9 pegs that could be sticked inside the holes. A small bowl was placed beside the board to grap the pegs out or place them back into.
In this test, the participant is required to pick up the pegs one by one and place them in the 9 different holes of the board. Once all nine holes are filled in, the participant has to pick the 9 pegs back up out of the board and place them back into the bowl, one by one. The participant needs to do all this in a single hand and as fast as they can do it. This test was also performed with both hands separately to see if there is a big difference between using your dominant hand and non-dominant hand. The time it took for the participant to complete this test is measured through a stopwatch.

B.2.2.2 Tap Test

The Tap test on the Futuro cube was a small test to let the user use different fingers on the Futuro cube and see if they had some hand dexterity to quickly switch between tapping with different fingers. The top of the cube would light up and the goal of the participant was to simply press the top 4 times, by using their index-, middle-, ring finger and pinky in that order, and then press any side of the cube with their thumb. They were asked to do this as fast as possible, letting the FC measure how long it took to recognize a tap with each finger, all in ms.

B.2.2.3 Flip Test

The Flip Test was a small test for the users' Pronation/Supination skills. They would be given the Futuro Cube, and their goal would be to quickly switch their arm from a supination position to a pronation position and visa versa. The FC would then measure how long it took for it to be flipped to the other side in ms.

B.2.2.4 Knock Test

The Knock test was a small test for the users' wrist skills. They would be required to hold their arm still and move their whole hand up and down by their wrists as if they were knocking on a door. The FC would once again measure how long it took for the whole movement to be completed in ms.

B.2.2.5 Rotate Test

The Rotate test was a small test again for detecting the users' finger dexterity. One side of the cube would light and their goal was to rotate the cube around so the lit-up side would be facing the top of the cube. Once the face was on the top side of the cube it would pick a new side and do this 6 times before considering a complete sequence. The only restriction the user had was that they were not allowed to move the cube with their wrists or hand, they were to rotate the cube by only using their fingers. The FC would measure how long it took to get to each side and how long it took to do a complete sequence, both in ms.

B.2.3 Experiment Results

All participants were right-handed, so their dominant hand was their right hand.

B.2.3.1 NPHT test

Each participant got times for both their dominant and non-dominant hand. No individual test took longer than a minute and the total test took was completed within 5 minutes, including explanation.

On average, performing the NPHT with their non-dominant hand before they played the game scored the best, with an average of 28.17 seconds to complete the test. It also had the fastest time of all tests with 22.77 seconds and the lowest slowest time with 40.23 seconds. The highest average of the tests is 32,96 seconds with the dominant hand after they played the game. The lowest fastest time is with the non-dominant hand after playing the game with 23.74 seconds and the highest slowest time is with the dominant hand before they played the game with 48.99 seconds.

On average, there is an improvement with both the dominant hand and the non-dominant after having played the game. The dominant hand is on average -4.05 seconds faster and the non-dominant hand is -1.62 seconds faster. Of the 29 participants, 23 were faster with their dominant hand, 5 were slower and 1 person was exactly the same. With their non-dominant hand, 18 participants were faster and 11 were slower.

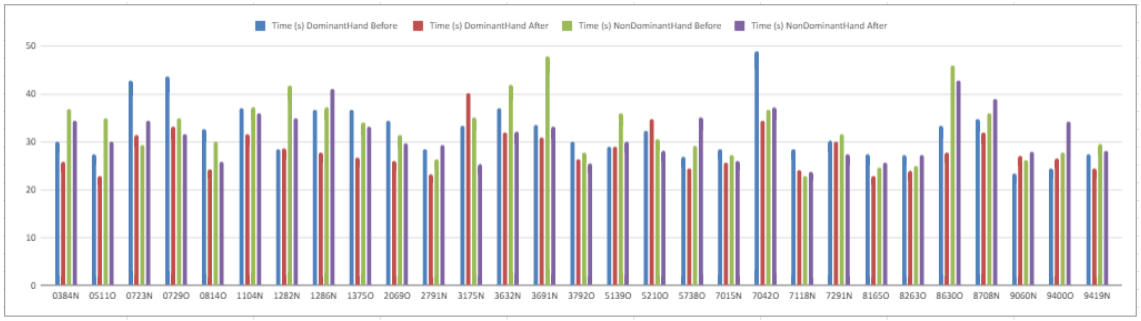


Figure B.3: The Average time in s it took to complete the Nine Peg Hole Test with both the dominant and non-dominant hand before and after the participant played the game

B.2.3.2 Tap Test

The participants were asked to do a tap sequence (of index, middle, ring finger, pink, and thumb) at least three times with each hand (dominant first). Each separate press was recorded. The index finger was used to establish the start of the test and would therefore return almost always the same value back and will be mostly ignored in the data analysis.

In the tap test before the participant played the game, the dominant hand was in general better, with 67 out of 87 (3*29) times the dominant hand being faster than the non-dominant hand. On average, the dominant hand is also faster with 2 of the 4 tested fingers (ring finger and pink). Also, the pink is shown to cause the most trouble with both the dominant and non-dominant hand, being the slowest by 20 participants with their dominant hand and 23 with their non-dominant hand. The fastest sequence was performed with the non-dominant hand, taking 520 ms, compared to the fastest tap sequence with the dominant hand being 576 ms. The slowest tap sequence was done with the dominant hand, taking 33264 ms (or 33 seconds) vs 14608 ms (or 14.5 seconds) with the non-dominant hand. A full sequence was on average better with the dominant hand taking 2025 ms compared to the 2831 ms with the non-dominant hand.
In the tap test after the participant played the game, the dominant hand is again better most of the time, this time 69 out of 87 times the dominant hand is faster than the non-dominant hand. On average, the dominant hand is this time faster with all the recorded fingers (middle and ring finger, and pink). After playing the game, the pink was again the finger that caused the most trouble, being on average the slowest by 15 out of 29 participants with their dominant hand and 21

out of 29 with their non-dominant hand. The fastest sequence was performed with the dominant hand, taking 352 ms, while the non-dominant hand's fastest time was 416 ms. The slowest tap sequence was still the dominant hand, with 11736 ms (or about 12 s) compared to the 9696 ms (or about 10 s) from the non-dominant hand's slowest tap sequence. A full sequence was on average better with the dominant hand taking 1341 ms compared to the 1926 ms with the non-dominant hand.

If we compare how well the participants score before and after the test, we see that only 27 cases of the 87 possible tested show an actual improvement in time with the dominant hand, and only 17 of the 87 with the non-dominant hand. For the fastest and slowest sequence, we can see that after playing the game both have become faster. As for the average of the full sequence, we see an increase of 33.74% with the dominant hand and a 31.95% increase with the non-dominant hand

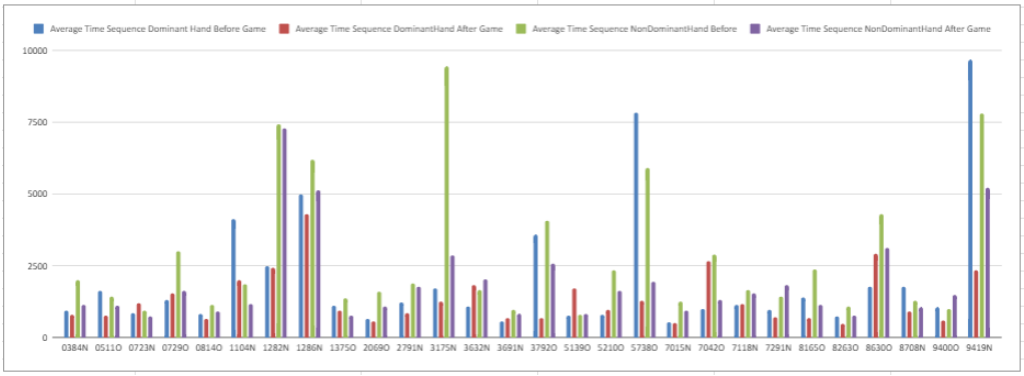


Figure B.4: The Average time in ms it took to complete a full tap sequence (index finger, middle finger, ring finger, pink and thumb) with both the dominant and non-dominant hand before and after the participant played the game

B.2.3.3 flip Test

The participants were asked to perform the flip test a total of 5 times with both hands. The system would regularly update itself to know which side was up and to no give the participant a too high flip time when they were still preparing to perform the test.

Before playing the game, we can see that the flip movement takes about 45 ms on average with the dominant hand and 53 ms with the non-dominant hand. Comparatively, after playing the game, the dominant hand takes 47 ms on average, while the non-dominant hand takes 58 ms. If we compare all 145 (5*29) entries of both the dominant and non-dominant hand, we see that for the dominant hand, 58 out of 145 (40%) show improvement, and for the non-dominant 67 out of 145 (46.2%) show improvement.

B.2.3.4 knock test

The participants were asked to perform the knock test a total of 5 times with both hands. The system would regularly update itself to know which side was up and to no give the participant a too high flip time when they were still preparing to perform the test.

Before playing the game, we can see that the knock movement takes about 61 ms on average with the dominant hand and 50 ms with the non-dominant hand. After having played the game, the dominant hand takes on average 49 ms to perform and the non-dominant hand takes 53 ms. If we compare all 145 entries of both the dominant and non-dominant hand, we see that for the

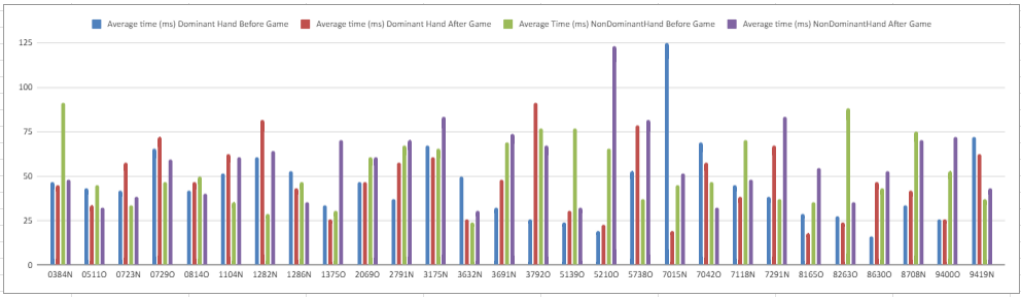


Figure B.5: The Average time in ms it took to Flip the Futuro Cube with both the dominant and non-dominant hand before and after the participant played the game

dominant hand, 52 out of 145 (35.8%) show improvement, and for the non-dominant hand 65 out of 145 (44.8%) show improvement.

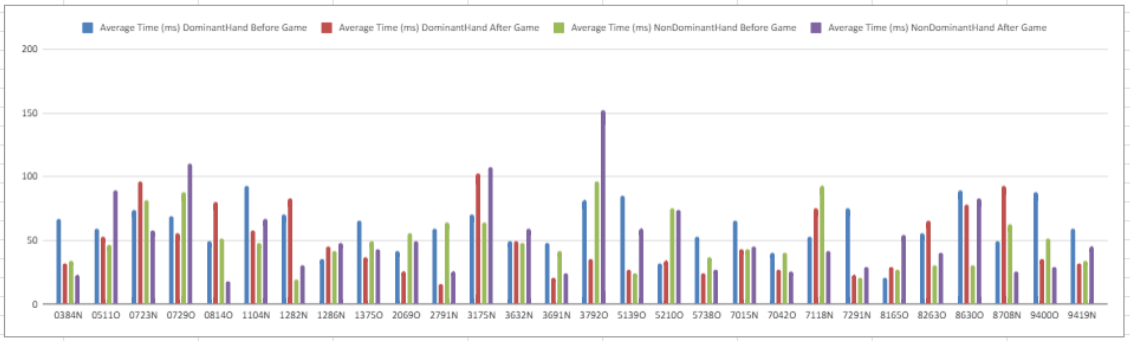


Figure B.6: The Average time in ms it took to make the knock motion with both the dominant and non-dominant hand before and after the participant played the game

B.2.3.5 rotate test

The participants were asked to perform a rotate sequence (rotating one side six times upwards with only your fingers) at least three times with each hand (dominant hand first). The first side rotation would be used to start recording the time and therefore resulted almost always the same value back and will be mostly ignored in the data analysis.

Before playing the game, with the dominant hand it takes an average 830 ms to rotate the cube to the correct side using only your fingers. On average, a sequence took 6781 ms with the fastest sequence taking 2528 ms and the slowest sequence taking 27040 ms. In the non-dominant hand, it takes 1216 ms on average to get to one side. For the full sequence, it takes on average 8012 ms, with the fastest time to complete a sequence taking 752 ms and the slowest sequence taking 19208 ms.
After playing the game, it takes an average of 663 ms to rotate the cube to the correct side using your dominant hand. On average, a sequence took 6781 ms with the fastest sequence taking 1648 ms and the slowest sequence taking 15280 ms. In the non-dominant hand, it takes 1095 ms on average to get to one side. For the full sequence, it takes on average 6447 ms, with the fastest time to complete a sequence taking 824 ms and the slowest sequence taking 10344 ms.
Looking at how often the participants performed better after playing the game after playing it compared to before, we see that 57 out of the 87 (65.5%) performed better with their dominant hand, and 62 out of the 87 (71.3%) performed better with their non-dominant hand. The average of a full sequence increased by 26.4% with the dominant hand and by 19.5% with the non-dominant hand.

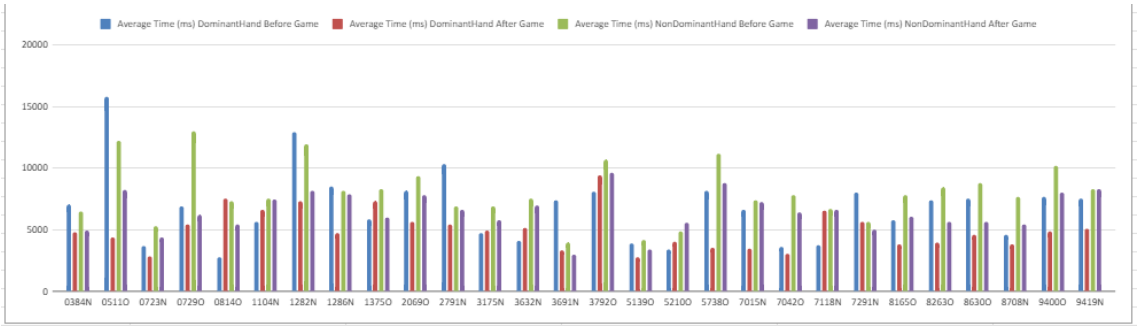
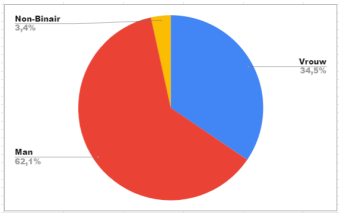


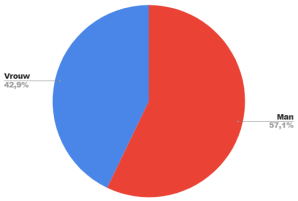
Figure B.7: The Average time in ms it took to complete a sequence where all 6 sides get rotated towards to top by only moving the Futuro Cube with the fingers of both the dominant and non-dominant hand before and after the participant played the game

B.2.3.6 Questionnnaire

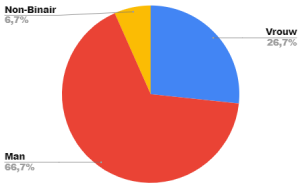
Of the 29 participants, 18 are male (62.1%), 10 are female (34.5%), and 1 (3.4%) is non-binary. 19 participants (65.5%) were in the range of being 21-25 years old, 3 participants were between 25-30 and 3 were older than 30 years (both 10.3%) and 4 participants were 18-21 years old (13,8%). 27 participants had no problems with their fine motor skills (93.3%) and 2 participants had problems with their fine motor skills, though one does not notice much of it in their daily lives, while the other does (both 3.3%). 27 participants did not have fine motor skill issues in their family (90%), while 2 participants (10%) have family members that have family members suffering from problems with their fine motor skills, not caused by simply becoming older. 12 participants (43.3%) walk once in a while to stay healthy, 7 participants (23,3%) actively play a weekly sport or go often to the gym, 5 participants (16.7%) do some small daily physical exercise, and 5 participants (16,7%) rarely go outside except when they need to. Playing Video games was the most often mentioned hobby the participants had, with 24 (82.8%) participants mentioning it. Other often mentioned hobbies are reading (6 participants or 20.7%), watching videos or shows (5 participants or 17.2%), listening or making music (4 participants or 13.8%), and playing boardgames (5 participants or 17.2%).



(a) Gender of all 29 participants

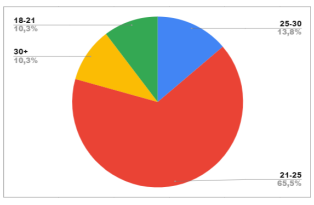


(b) Gender of all 14 participants testing the old game version

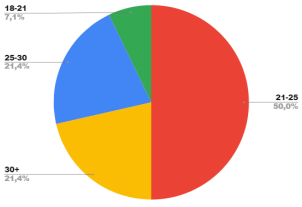


(c) Gender of all 15 participants testing the new game version

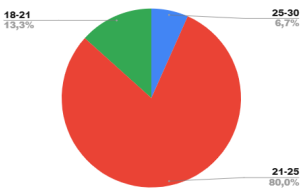
Figure B.8: Graphs of the representation of gender over the experiment.



(a) Age range of all 29 participants

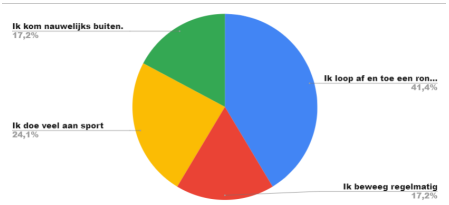


(b) Age range of all 14 participants testing the old game version



(c) Age Range of all 15 participants testing the new game version

Figure B.9: Graphs of the representation of the age-ranges over the experiment.



(a) Activity Measure of all 29 participants

(b) Activity Measure of all 14 participants testing the old game version

(c) Activity Measure of all 15 participants testing the new game version

Figure B.10: Graphs of the representation of how active the participants of the experiment are.

Looking at the 13 flow statements, we see that most participant:

- Felt the right amount of challenge
- did not notice time passing
- Had no trouble concentrating on the game
- mind was completely clear
- completely absorbed in the game
- know what actions they need to take

If we compare the two versions we see that:

- Believed that the game gave just the right amount of challenge more with the old version than with the new version
- Playing the old version played smoother than playing the new version
- Both versions made the participants lose their track of time equally
- In the old version made it easier to concentrate on what you were doing than the new version.
- In both versions caused the participants' minds to become clear.
- In both the new and old versions the participants got absorbed in the game
- The movements made with the old version felt more natural than the movements made with the new version
- it was more clear what the participant needed to do in the old version of the game than in the new version.
- the participants felt more in control playing the old version than playing the new version.
- Playing the new version made the participants lost in their thoughts more than playing the old version
- Both versions were considered equally important to the participants
- Participants forced themselves more to not make mistakes with the new version compared to playing the old version
- The participants were more scared of failing in the old version than in the new version.

All the data can also be found in Figures B.12, B.13 and B.14.

The participants were also allowed to give some additional feedback at the end of everything. The comments go in different directions, but the most common complaints were about Rotating and Snake minigames. Snake was considered very frustrating to players because it was very difficult for them to get the Snake moving the way they wanted, either because they tapped wrong or because the cube did not register the tap. Furthermore, sometimes if you are rotating the cube to have to correct side facing up, the snake could register it as tapping the side of the cube, which could lead the Snake to move away if the participant didn't want to. Rotate was disliked for two reasons. The first reason is that it was not clear to the participants which way the lights were rotating. It could take sometimes participants too long to realize which way they were supposed to rotate, that the minigame already failed before they had a chance to perform it. Additionally, the rotation action would not always be recognized, making the participants unsure if they were performing the rotation correctly. This would be added to the fact that some participants rotated the wrong way with the cube. Because of the way the player walks over the cube, the player will move for some participants in the opposite way as they would expect. In the rotate minigame you're supposed to let the Player follow the rotating lights, but that means you have to rotate the cube in the opposite direction as the lights move.

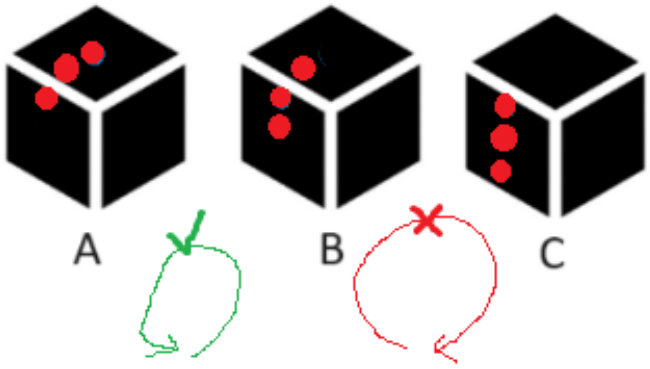


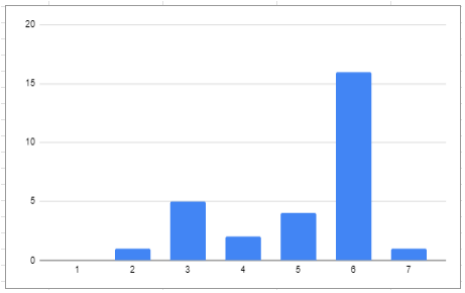
Figure B.11: A–>B–>C shows how the dots rotate over the cube. If you would rotate the Futuro Cube the same way the dots would be moving, it would the wrong way for the player to move that way.

B.2.3.7 The Game Data

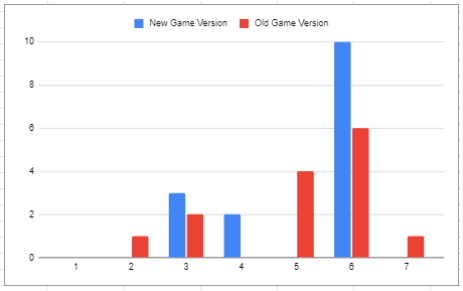
The 29 participants have played a grand total of 2318 separate minigames. Of these minigames, 1683 were completed successfully and 635 minigames ended in failure. The game data of one participant ended up corrupted, losing 119 (59 successes, 60 fails) data points.
Table B.6 and Figure B.15 give a short overview the minigame data.

Find The dot success rate decreases by 10% and Tap the side by 30%, while Shake increases to 100% (increasing about 18%) and Rotate doubling in its success rate (increasing with about 42%). Flip and Special Shake show a good win rate with about 77.1% and 72.1% win rate. Also, Road-Runner scores relatively well, scoring 54.9%, meaning users have a slightly more likely chance to complete it than not. Snake scores the worst from both the new set games and all Success rates of the New Version, with 43,7%.

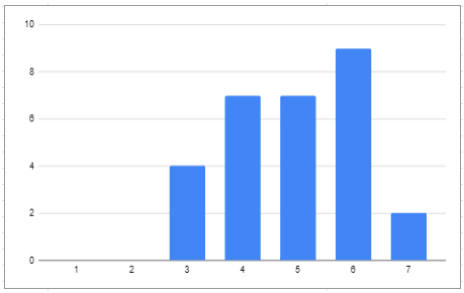
Figures B.16, B.17 and B.18 show a more worked out win-lose ratio on all the different AV-levels. Figure B.16 shows that in the old version of the game, we only have a change of failure on AV levels 1 and 4. This is mostly due to the way the old code changes the AV. This causes the AV to not often be 2 or 3, leading to no failures during the experiment. An additional thing to be noted is the lack of other minigames on AV 1. This can be explained with Algorithm 2. A lower AV disables the "more difficult minigames". Meaning on AV level 1, only Find the dot can be chosen.



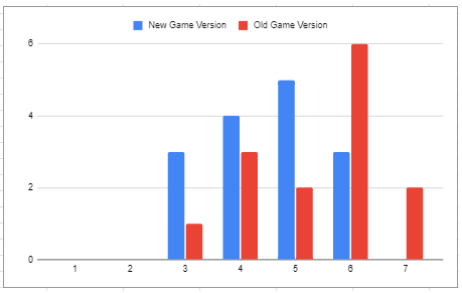
(a) "I feel just the right amount of challenge"



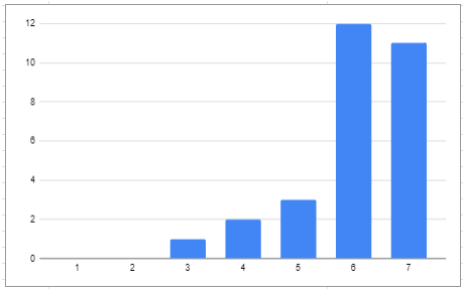
(b) "I feel just the right amount of challenge"



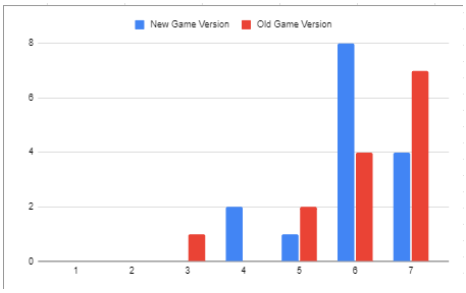
(c) "My Thoughts/activities run fluidly and smoothly"



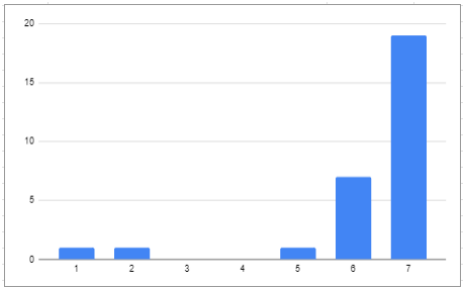
(d) "My Thoughts/activities run fluidly and smoothly"



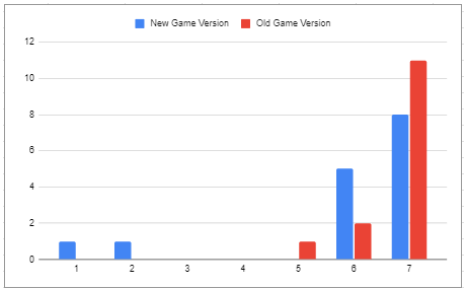
(e) "I don't notice time passing"



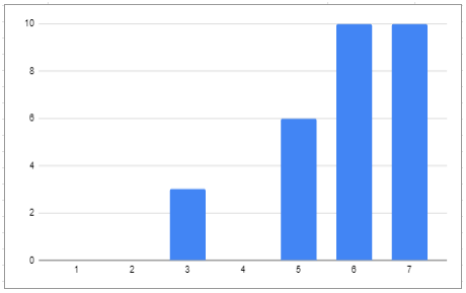
(f) "I don't notice time passing"



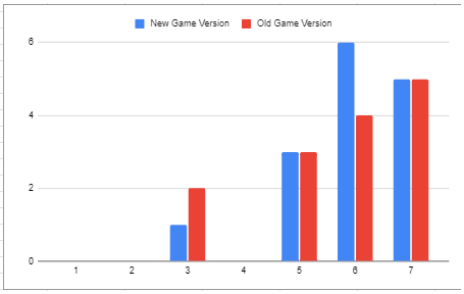
(g) "I have no difficulty concentrating"



(h) "I have no difficulty concentrating"

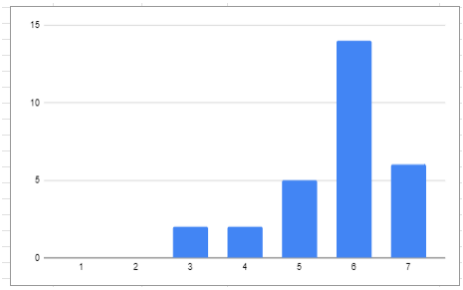


(i) "My mind is completely clear"

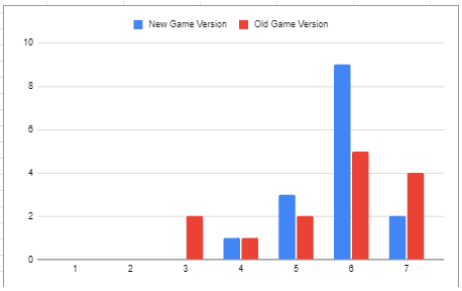


(j) "My mind is completely clear"

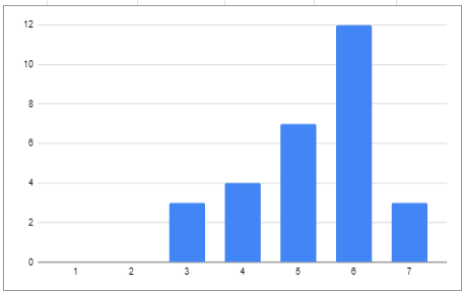
Figure B.12: Results from the flow short scale, to the left the total result of the whole experiment, to the right the comparison between the two versions. The caption below is the statement the participant answered about how strongly they disagree (1) or agree (7)



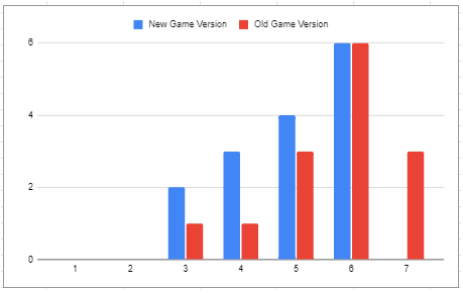
(a) "I am totally absorbed in what I am doing"



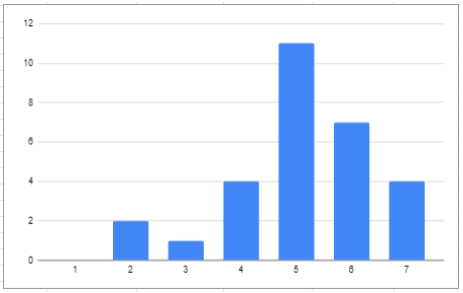
(b) "I am totally absorbed in what I am doing"



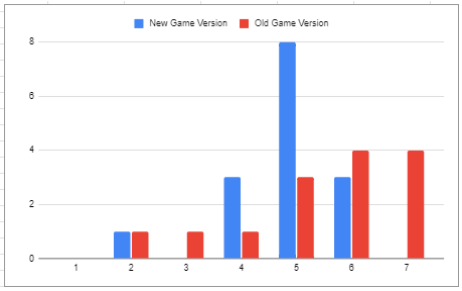
(c) "The right thoughts/movements occur of their own accord"



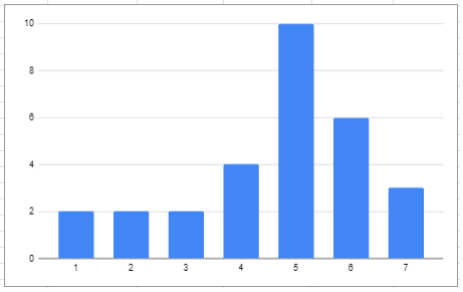
(d) "The right thoughts/movements occur of their own accord"



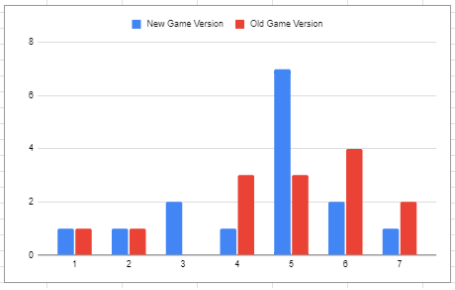
(e) "I know what I have to do each step of the way"



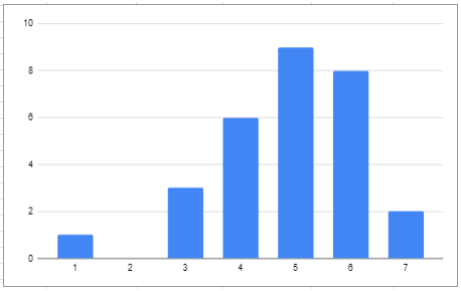
(f) "I know what I have to do each step of the way"



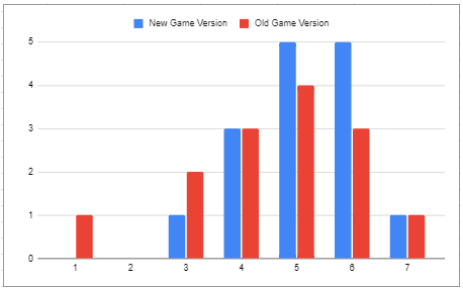
(g) "I feel that I have everything under control"



(h) "I feel that I have everything under control"

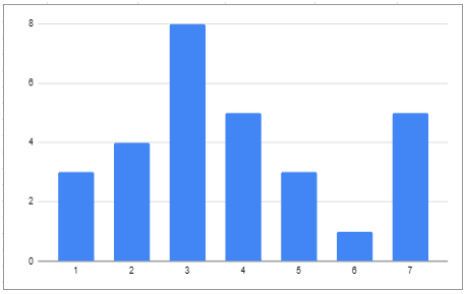


(i) "I am completely Lost in thought"

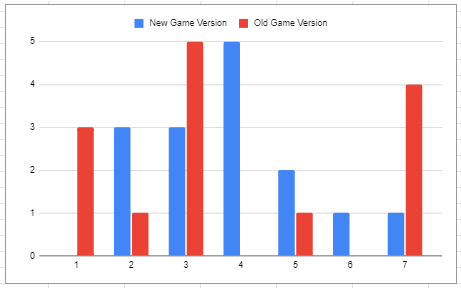


(j) "I am completely Lost in thought"

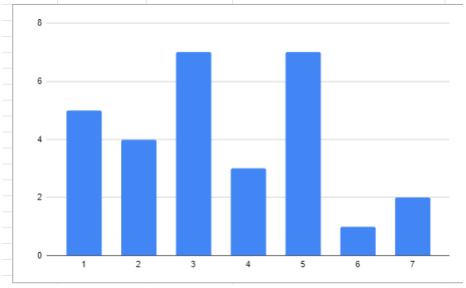
Figure B.13: Results from the flow short scale, to the left the total result of the whole experiment, to the right the comparison between the two versions. The caption below is the statement the participant answered about how strongly they disagree (1) or agree (7)



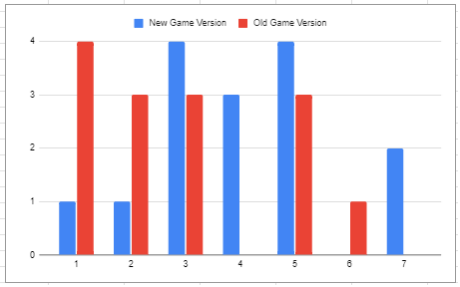
(a) "Something important to me is at stake here"



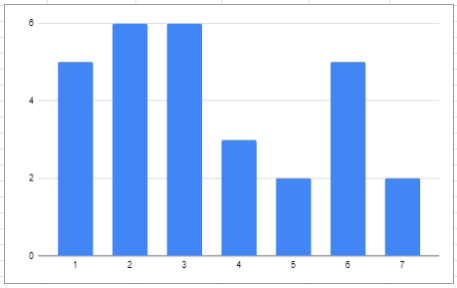
(b) "Something important to me is at stake here"



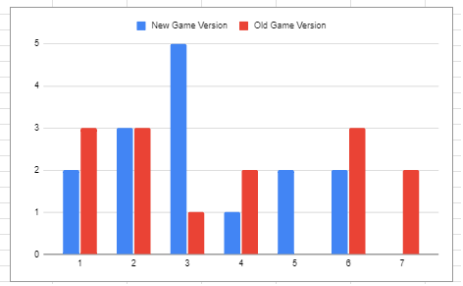
(c) "I won't make any mistakes here"



(d) "I won't make any mistakes here"

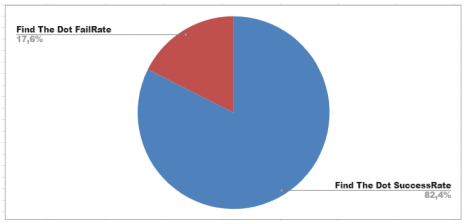


(e) "I am worried about failing"

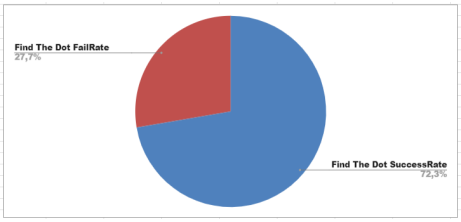


(f) "I am worried about failing"

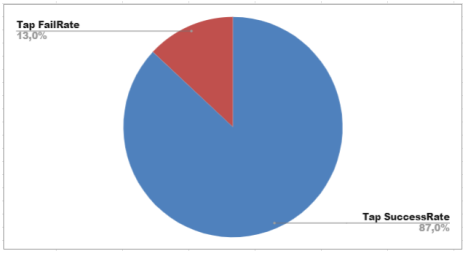
Figure B.14: Results from the flow short scale, to the left the total result of the whole experiment, to the right the comparison between the two versions. The caption below is the statement the participant answered about how strongly they disagree (1) or agree (7)



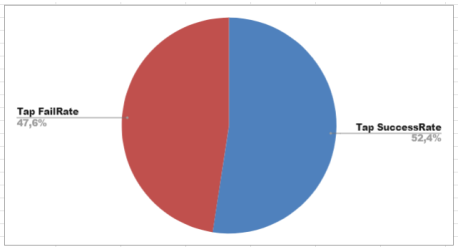
(a) SuccessRate of Find the Dot in the old game version



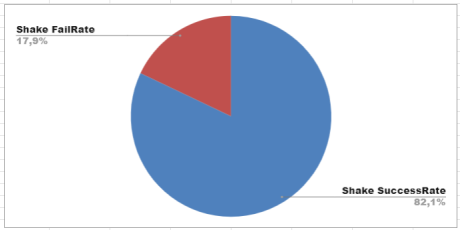
(b) SuccessRate of Find the Dot in the new game version



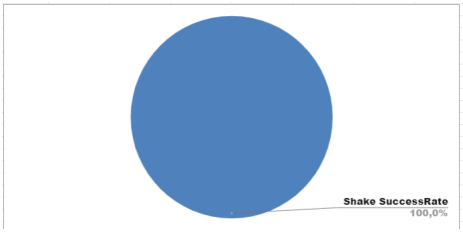
(c) SuccessRate of Tap in the old game version



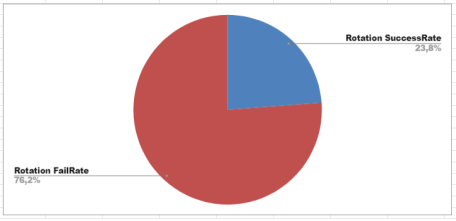
(d) SuccessRate of Tap in the new game version



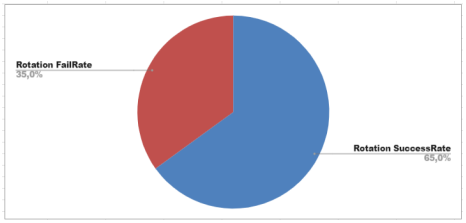
(e) SuccessRate of Shake in the old game version



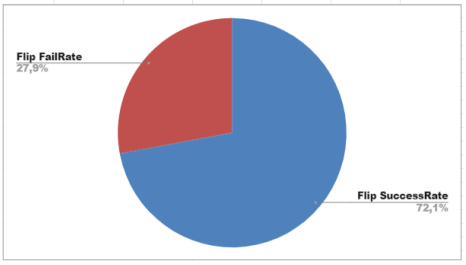
(f) SuccessRate of Shake in the new game version



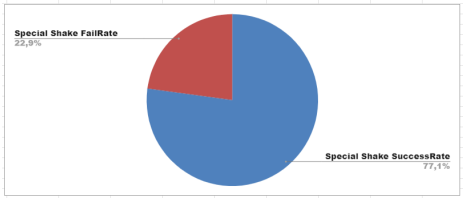
(g) SuccessRate of Rotate in the old game version



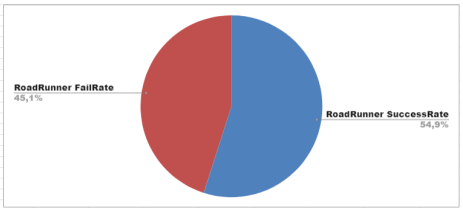
(h) SuccessRate of Rotate in the new game version



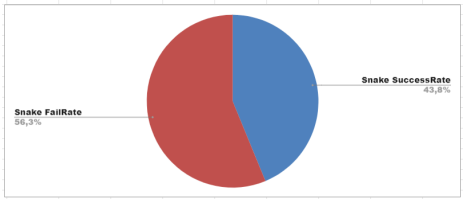
(i) SuccessRate of Flip in the new game version



(j) SuccessRate of Special Shake in the ne game version



(k) SuccessRate of RoadRunner in the new game version



(l) SuccessRate of Snake in the new game version

Figure B.15: The SuccessRate of all the minigames, also comparing the old with the new versions

Minigame	Version	Successes	Fails	Success Rate
Find The dot	Old Version	627	134	82,4%
Find The dot	New Version	512	196	72,3%
Tap	Old Version	80	12	87,0%
Tap	New Version	54	49	52,4%
Shake	Old Version	32	7	82,1%
Shake	New Version	53	0	100%
Rotate	Old Version	10	32	23,8%
Rotate	New Version	67	36	65,0%
Flip	New Version	62	24	72,1%
Special Shake	New Version	27	8	77,1%
RoadRunner	New Version	39	32	54,9%
Snake	New Version	35	45	43,7%

Table B.6: The success of each minigame, also split between the old and new game verions

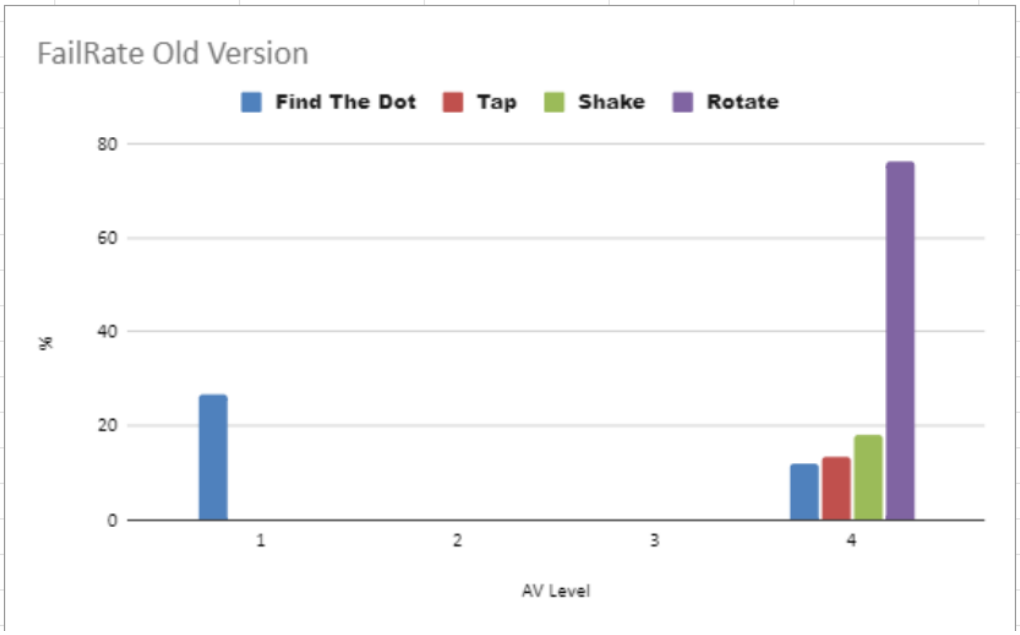


Figure B.16: How likely (in %) the user is to lose a minigame on each of the different minigames in the old version (based on the results of the experiment)

This also in turn means that the success rate of all the other minigames are non-applicable, for they are not played on this AV level.
For the new version of the game, we see very interesting results in Figure B.17. We can see that the fail rate of Find the dot is somewhat stable from AV level 6-8 and even decreases. From AV levels 9 and 10, it very quickly increases. Tap Shows a mostly increase in fail rate, except for AV level 8. Shaking is not present in this graph, for its success rate during the experiment was 100%. Rotate has a higher fail rate on the lower AV levels, than on the higher AV rates. Still, both AV 6-7 and 8-10 show an increase in fail rate. Flip shows to be fluctuating over the different AV levels but staying mostly stable. Special Shake shows that on higher AV levels players seem to be more likely to succeed than on earlier levels. Special shake is absent on AV level 10 not due to it having a 100% success rate but due to it never appearing while the user was on the highest AV level. RoadRunner fluctuates over the different AV levels but stays mostly stable. Snake shows to mostly become more difficult towards a higher AV Level, though from AV level 8 it did seem to become slightly easier for most users.

For the old game version, the average AV level of the participants over the game is 2.96. The

average AV level the participants ended with is 2.5, which is a difference of 0.46. The average difference between the average AV level the users experienced during the game and the AV level they ended with is 1.41, with the biggest difference being 2.26 and the smallest difference 0.80. Furthermore, if we look at which AV level the participants played the longest on, we see that all 14 participants played the longest time on AV level 4. Still, half of the participants (7) ended with an AV level of 1.
For the new game version, the average AV level of the participants over the game is 8.11. The average AV level the participants ended with is 8.21, which is a difference of 0.1. The average difference between the average AV level of the users experienced during the game and the AV level they ended with is 0.23, with the biggest difference being 0.81 and the smallest difference 0.01. Also, if we look at which AV level the participants played the longest on, we see that 7 participants played on the second highest AV level (9) the longest, 4 participants on AV level 8 the longest, 2 participants on level 7 the longest and one participant on level 6. On top of that, each participant ended with the AV level, they played the longest on.

Lastly, we looked at the average amount of leftover time the user had after completing the minigame successfully. This was done to see if higher AV levels would show that it was more difficult to complete it in time (i.e. have less time over).

For the old game version, we have sadly limited game data, but we still can see that in most cases it does show a downward direction, so it suggests that it becomes more difficult to complete with a higher AV level.
For the new game version, we see that Find the Dot, Tap, and Rotate indeed have less time after the player has completed it. Shake and Flip do show that at higher AV levels the remaining time goes down, however, they also show a heavy increase going from AV level 5 to 6. Snake stays very steady around the same values through all the different AV levels and Special Shake shows an increase in time left, meaning that the game becomes easier on higher AV levels.
If we compare the old and the new versions, then for Find the Dot and Tap, the new version has less time over than the old version. For Shake and Rotate it shows that on the max AV level, the new version has about as much time over as the old version, while on lower AV levels the participants often had more time over. you can see these graphs in Figure B.20

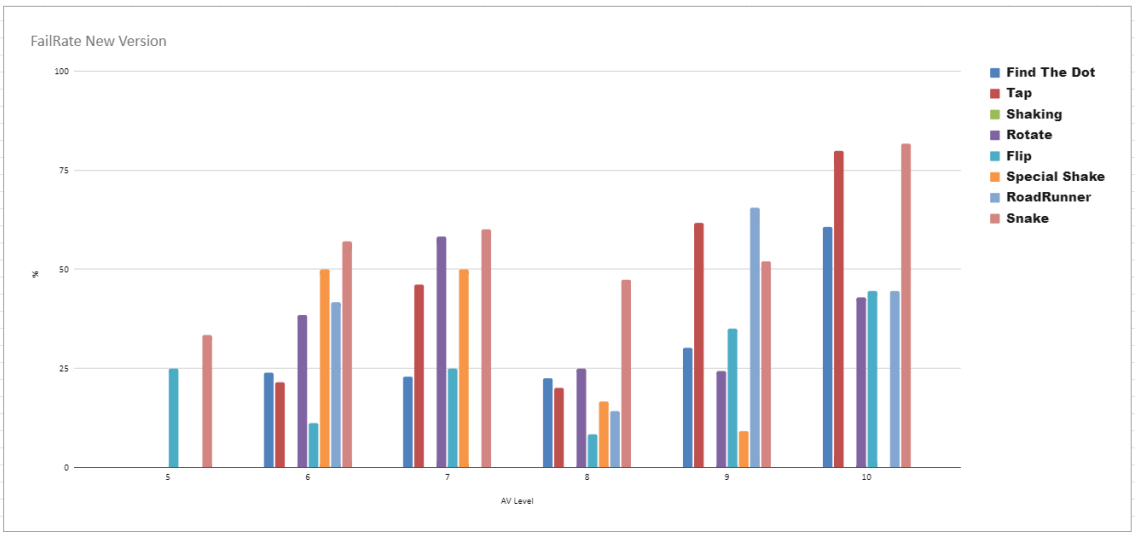
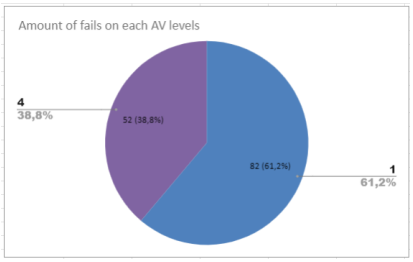
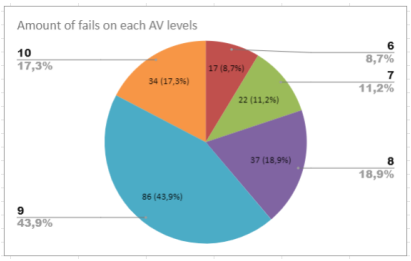


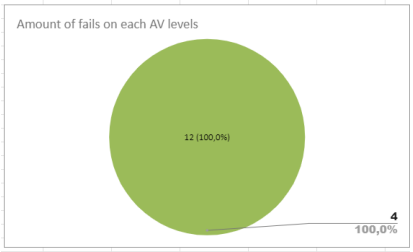
Figure B.17: How likely (in %) the user is to lose a minigame on each of the different minigames in the new version (based on the results of the experiment)



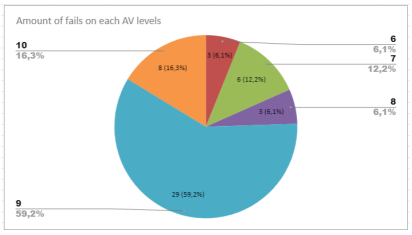
(a) Fail Distribution of Find the Dot in the old game version



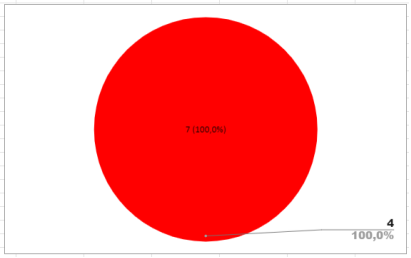
(b) Fail Distribution of Find the Dot in the new game version



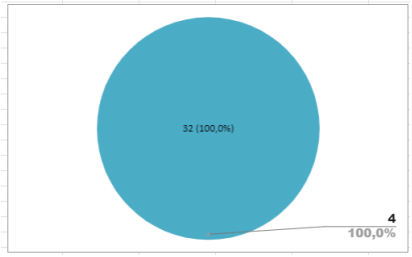
(c) Fail Distribution of Tap in the old game version



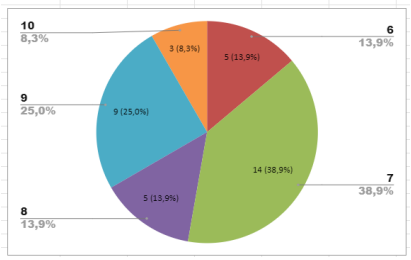
(d) Fail Distribution of Tap in the new game version



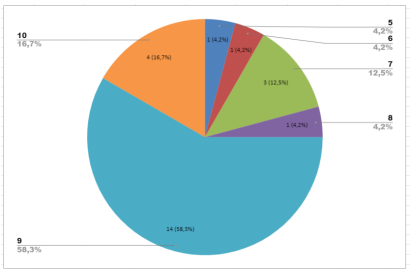
(e) Fail Distribution of Shake in the old game version



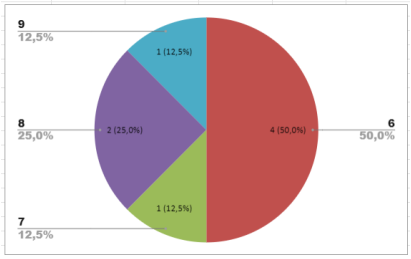
(f) Fail Distribution of Rotate in the old game version



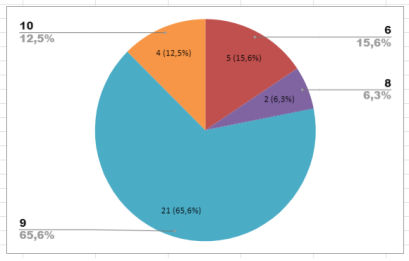
(g) Fail Distribution of Rotate in the new game version



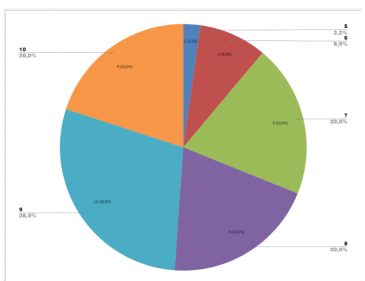
(h) Fail Distribution of Flip in the new game version



(i) Fail Distribution of Special Shake in the new game version

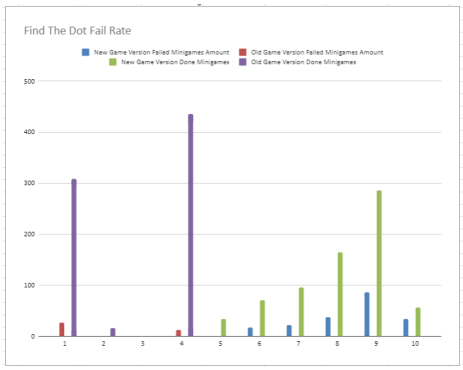


(j) Fail Distribution of RoadRunner in the new game version

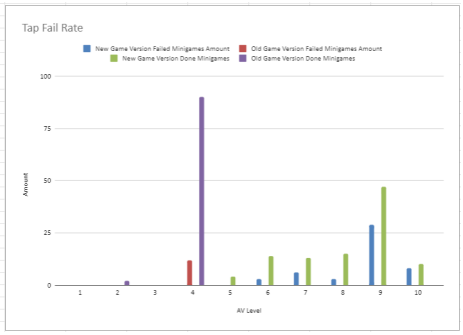


(k) Fail Distribution of Snake in the new game version

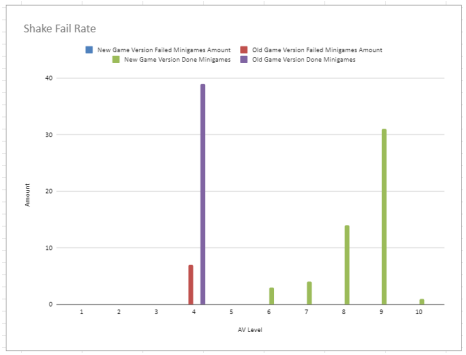
Figure B.18: The distribution of the fails over the different AV Levels



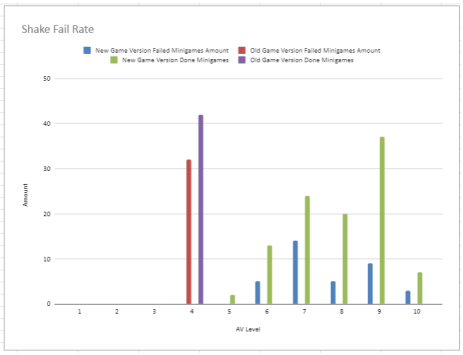
(a) FailRate of Find the Dot



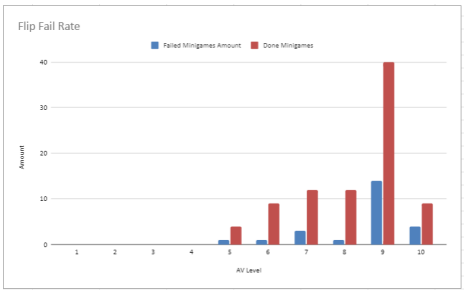
(b) FailRate of Tap



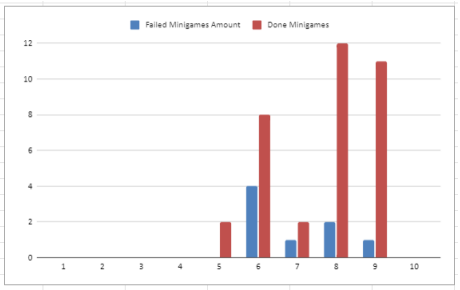
(c) FailRate of Shake



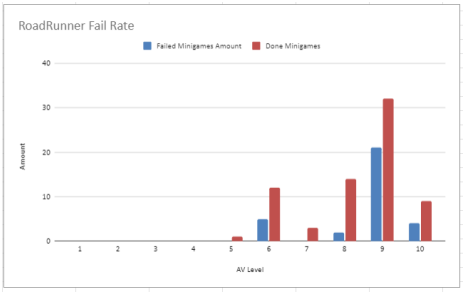
(d) FailRate of Rotate



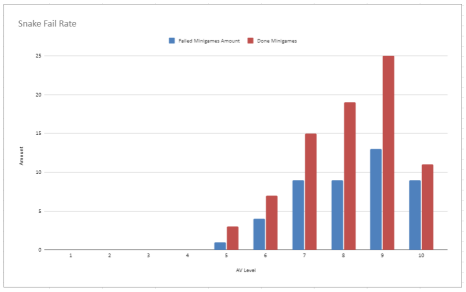
(e) FailRate of Flip



(f) FailRate of Special Shake

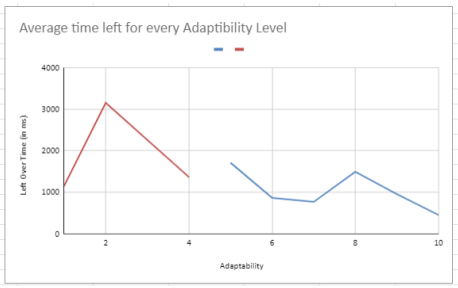


(g) FailRate of RoadRunner

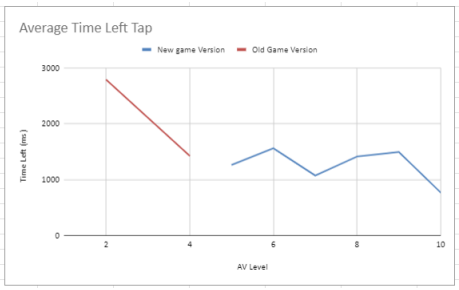


(h) FailRate of Snake

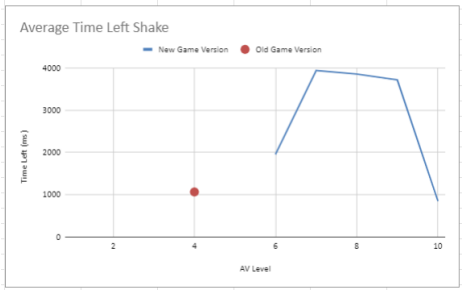
Figure B.19: The FailRate of all the minigames by showing the amount of failures with the amount of played games, also comparing the old with the new versions



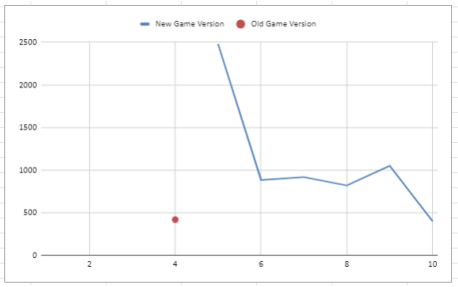
(a) Average Time Left for Find the Dot



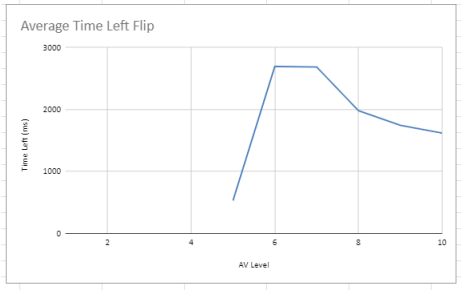
(b) Average Time Left for Tap



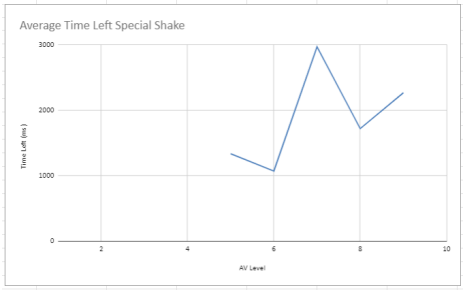
(c) Average Time Left for Shake



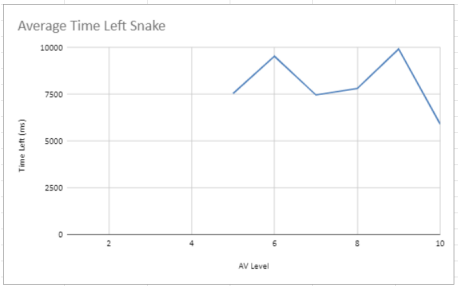
(d) Average Time Left for Rotate



(e) Average Time Left for Flip



(f) Average Time Left for Special Shake



(g) Average Time Left for Snake

Figure B.20: The Average Time Left for all the minigames, also comparing the old with the new versions

B.3 Statistical Analysis

Finally, we will perform some statistical analysis on some of the data to see if there is significant data to detect. There are 3 things we want to analyze:

1. Is there a significant improvement in fine motor skills before and after playing with the Futuro Cube
2. Is playing the minigames on the new Futuro Cube version significantly more difficult?
3. Do the participants experience flow while playing with the Futuro Cube?

B.3.1 Fine Motor Skill Improvement

Because of the short amount of time we play with the FC we don't expect an increase in Fine Motor skills for the user to play before or after they play with the FC. this means are null hypothesis is as follows:
H_0: *There is no difference in fine motor skills before and after playing with the Futuro Cube*
In which we will have the following Alternative Hypothesis:
H_A: *There is a difference in fine motor skills before and after playing with the Futuro Cube*
We are interested in knowing whether we detect a significant difference in fine motor skills because this would mean that there is a variable that has had an influence on the test results. Additionally, we have that we have tested the same person multiple times with the same test. This means that we can perform a two-tailed two-sampled T-Test. To decrease the amount of data processing, we will pick the average difference of both dominant and non-dominant hands. Additionally, one participant's data got lost, so except for the NPHT test where N = 29, the other tests use N = 28

B.3.1.1 NinePegHoleTest

the mean is $\overline{d} \approx -2.831$ and the sum of all squares is 254.61. This means the standard deviation is: $\sqrt{\frac{254,61}{29-1}} \approx 9,09$

we have t-statistic of $\frac{-2.831}{\frac{9.09}{\sqrt{29}}} \approx -1,68$, a degree of freedom of $29-1=28$ and a $t_{critical} = 2.48$ following the two-tails T Distribution table [39], we see that our t-statistic < $t_{cricital}$, which indicates that we don't have a significant difference in fine motor skill before and after performing the test. In other words the Nine Peg Hole test does indeed not show a difference in fine motor skills.

B.3.1.2 Tap

The mean is $\overline{d} \approx -793,876$ and the sum of all squares is 58059139. This means our standard deviation is $\sqrt{\frac{58059139}{28-1}} \approx 1466,4$
This means we have a t-statistic of $\frac{-793,876}{\frac{1466,4}{\sqrt{28}}} \approx -2,865$ and a degree of freedom of $28-1=27$. This results in a $T_{critical} = 2,052$ Because our T-statistic is higher, we do see a significant difference before and after playing the game in fine motor skills if we look at the Tap test.

B.3.1.3 Flip

The mean is $\overline{d} \approx 3.43$and the sum of all squares is 11210. This means the standard deviation is: $\sqrt{\frac{11210}{28-1}} \approx 20,376$. This results in a t-statistic of $\frac{3,43}{\frac{20,376}{\sqrt{28}}} \approx 0,89$ and a degree of freedom of $28-1=27$. This results in a $T_{critical} = 2,052$. Our T-statistic is far smaller than our critical T-value so we accept the Null hypothesis when looking at the flip test.

B.3.1.4 Knock

our mean is $\overline{d} \approx -1677,33$ and the sum of all squares is 10140525. This means the standard deviation is: $\sqrt{\frac{10140525}{28-1}} \approx 612,842$. This results in a t-statistic of $\frac{-1677,33}{\frac{612,842}{\sqrt{28}}} \approx -1,276$ and a degree of freedom of $28 - 1 = 27$. This results in a $T_{critical} = 2,052$. Our T-statistic is far smaller than our critical T-value so we accept the Null hypothesis when looking at the knock test.

B.3.1.5 Rotate

The mean is $\overline{d} \approx -1677,33$ and the sum of all squares is 11210. This means the standard deviation is: $\sqrt{\frac{11210}{28-1}} \approx 20,376$. This results in a t-statistic of $\frac{-1677,3}{\frac{20,376}{\sqrt{28}}} \approx -14,483$ and a degree of freedom of $28 - 1 = 27$. This results in a $T_{critical} = 2,052$. Because our T-statistic is higher than our critical T-value shows rotate that there is a significant difference in the fine motor skills of the users before and after playing with the Futuro Cube.

B.3.2 More Difficult minigames

We want to specifically check whether the new DDA system is better than the old one making the minigames more difficult. We want to see if making the minigames themselves more difficult means that the participants had more trouble completing the minigames than if we just adjusted the time for it. So we have:
H_0: *The new DDA system does not make it more difficult than the old version*
H_A: *The new DDA system is more difficult than the old version.*
Because we let participants play two different versions, we have that the samples are independent, and we are only interested in seeing if the new version is more difficult, so we can perform an independent (two) sampled one-tail t-test, to see if there is a significant difference.
Because we're going to compare the success rates of the different versions and we want to check if the new version is more difficult, we want to check whether $\mu_{NewVersion} < \mu_{OldVersion}$. For Find the Dot and Tap, we can look at the increase of fail rate (or decrease in success rate) to check for a significant increase. Shake and Rotate are both in the old version and only appear on AV level 4 in our data, so we will just compare the average fail rate with each other. For the new minigames, we will make a comparison between the overall fail rate of the new version, with the overall fail rate of the old version.

As stated before, one participant's test results got corrupted, so we have $N_{new} = 14$ and $N_{old} = 14$, though some minigames did not appear in each version. When this is the case it will be specified in the subsection.

B.3.2.1 Find the Dot

We have the following means for the success rates increase/decrease of the find the dot minigame over the two versions:
$\overline{X}_{NewVersion} = -17,44$
$\overline{X}_{OldVersion} = 15,07$
and we have the following standard deviations:
$\sigma_{NewVersion} = \sqrt{\frac{736.6}{14-1}} \approx 7,53$
$\sigma_{OldVersion} = \sqrt{\frac{2431,48}{14-1}} \approx 13.68$
$\sigma_{pool} = \sqrt{\frac{(14-1)*(7.53)^2+(14-1)*(13.68)^2}{14+14-2}} \approx 11.04$
This gives us a standard Error (SE) of: $SE = 11.04\sqrt{\frac{1}{14}+\frac{1}{14}} \approx 4,17$ and thus a t-statistic of $t = \frac{-17.44-15.07}{11.04} \approx -2.94$.
We have a degree of freedom of 14 + 14 - 2 = 26. Which means we would have a p-value of < 0.005. This means that the new find the dot is indeed more difficult than the original version.

B.3.2.2 Tap

We have the following means for the success rate increase/decrease of the Tap Minigame over the two versions:
$\overline{X}_{NewVersion} = -10,23$
$\overline{X}_{OldVersion} = 8,25$
and we have the following standard deviations:
$\sigma_{NewVersion} = \sqrt{\frac{11248,9}{14-1}} \approx 29.42$
$\sigma_{OldVersion} = \sqrt{\frac{13819,05}{14-1}} \approx 32.6$
$\sigma_{pool} = \sqrt{\frac{(14-1)*(29.42)^2+(14-1)*(32.6)^2}{14+14-2}} \approx 31.05$
This gives us a SE of: $SE = 31.05\sqrt{\frac{1}{14}+\frac{1}{14}} \approx 11.74$ and thus a t-statistic of $t = \frac{-10,23-8.25}{11.74} \approx -1.57$.
We have a degree of freedom of 14 + 14 - 2 = 26. Which means we would have a p-value of > 0.05. This means that the new Tap the side is not more difficult than the original version.

B.3.2.3 Shake

In the old version, the shake minigame did not occur once during the experiment. In the new version, the shake minigame did not appear for two of the experiments. This results in a $N_{new} = 12$ and $N_{old} = 13$ We have the following means for the success rates of the shake Minigame over the two versions:
$\overline{X}_{NewVersion} = 100$
$\overline{X}_{OldVersion} = 82.05$
and we have the following standard deviations:
$\sigma_{NewVersion} = \sqrt{\frac{0}{12-1}} = 0$
$\sigma_{OldVersion} = \sqrt{\frac{12478,63}{13-1}} \approx 30.98$
$\sigma_{pool} = \sqrt{\frac{(12-1)*(0)^2+(13-1)*(32.6)^2}{12+13-2}} \approx 23.55$
This gives us a SE of: $SE = 31.05\sqrt{\frac{1}{12}+\frac{1}{13}} \approx 9,43$ and thus a t-statistic of $t = \frac{100-82.05}{9,43} \approx 1.90$.
This is a positive number, which means that the game has become easier, not harder, so the alternative hypothesis is false for the Shake minigame

B.3.2.4 Rotate

In the new version, the rotate minigame did not occur during one of the experiments, which means we have the following N_{new} and N_{old}:
$N_{new} = 13$ and $N_{old} = 14$
We have the following means for the success rates of the Rotate Minigame over the two versions:
$\overline{X}_{NewVersion} = 65.42$
$\overline{X}_{OldVersion} = 23.81$
and we have the following standard deviations:
$\sigma_{NewVersion} = \sqrt{\frac{5815,41}{13-1}} \approx 22.01$
$\sigma_{OldVersion} = \sqrt{\frac{7619.05}{14-1}} \approx 24.21$
$\sigma_{pool} = \sqrt{\frac{(13-1)*(22.01)^2+(14-1)*(24.21)^2}{13+14-2}} \approx 23.18$
This gives us a SE of: $SE = 23.18\sqrt{\frac{1}{13}+\frac{1}{14}} \approx 8,93$ and thus a t-statistic of $t = \frac{65.42-23.81}{8,93} \approx 4,66$.
This is a positive number, which means that the game has become easier, not harder, so the alternative hypothesis is false for the Rotate minigame

B.3.2.5 New Games

Finally, we will take a look at the games overall. First we look at an even comparison, so we look at the success rate of only the minigames find the dot, tap, rotate, and shake. For this small set we have a mean success rate of: $\overline{X}_{NewVersion} = 72.26$
$\overline{X}_{OldVersion} = 81.18$
and we have the following standard deviations:
$\sigma_{NewVersion} = \sqrt{\frac{901.88}{14-1}} \approx 8,33$
$\sigma_{OldVersion} = \sqrt{\frac{789,65}{14-1}} \approx 7.79$
$\sigma_{pool} = \sqrt{\frac{(14-1)*(8.33)^2+(14-1)*(7.79)^2}{14+14-2}} \approx 8.06$
This gives us a SE of: $SE = 8.06\sqrt{\frac{1}{14}+\frac{1}{14}} \approx 3.05$ and thus a t-statistic of $t = \frac{72.26-81.18}{3.05} \approx -2.92$. This results in a p-value < 0.005. This means that overall, the new minigames are experienced as more difficult.

We do have the overall success and failrate of the data that got corrupted, so for the following test only, we can use $N_{new} = 15$ and $N_{old} = 14$
If we then also look at the total comparison between the two versions, we see a mean success rate of: $\overline{X}_{NewVersion} = 68,19$
$\overline{X}_{OldVersion} = 81,18$
and we have the following standard deviations:
$\sigma_{NewVersion} = \sqrt{\frac{1064,94}{15-1}} \approx 8.72$
$\sigma_{OldVersion} = \sqrt{\frac{789.65}{14-1}} \approx 7.79$
$\sigma_{pool} = \sqrt{\frac{(15-1)*(8.72)^2+(14-1)*(7.79)^2}{15+14-2}} \approx 8.29$
This gives us a SE of: $SE = 8.29\sqrt{\frac{1}{15}+\frac{1}{14}} \approx 3.08$ and thus a t-statistic of $t = \frac{68,19-81.18}{3.08} \approx -4.22$. with a degree of freedom of 27, this means we have a p-value < 0.0005, so compared to all the games, the new version is experienced as more difficult.

As a final comparison, we will look at the difference between series 2 (the new versions of find the dot, tap, shake and rotate) and 3 (all minigames). There we have the following means: $\overline{X}_{Series3} = 68,19$ $\overline{X}_{Series2} = 72.26$
with the following standard deviations:
$\sigma_{Series3} = \sqrt{\frac{1064,94}{15-1}} \approx 8.72$
$\sigma_{NewVersion} = \sqrt{\frac{901.88}{14-1}} \approx 8,33$
$\sigma_{pool} = \sqrt{\frac{(15-1)*(8.72)^2+(14-1)*(8.33)^2}{15+14-2}} \approx 8.53$
This gives us a SE of: $SE = 8.53\sqrt{\frac{1}{15}+\frac{1}{14}} \approx 3.16$ and thus a t-statistic of $t = \frac{68,19-81.18}{3.08} \approx -1.28$. With a degree of freedom of 27, this means that we have a p-value of >0.1, so there is no significant improvement when we look at the difference between minigameserie2 and minigameserie3.

B.3.3 Better Flow Experience

This is on its own separated into two separate questions: Do the players experience Flow at all while playing with the Futuro Cube and do the players experience better flow while playing the new version than the older version or the other way around.

B.3.3.1 Flow Game

To see if we enter a state of flow with our game we can use the provided norm tables in the paper of FSS[33], see also Figure B.21 and Table B.7.

Experiment Number	Flow Questions	$T_{Flow}Score$	Risk Questions	$T_{Worry}Score$
9060N	53	52	12	56
5139O	53	52	9	56
3175N	56	55	7	52
1104N	55	54	10	48
3691N	59	58	17	53
8708N	61	60	14	63
0723N	43	45	8	59
5210O	60	59	18	46
2069O	56	55	10	64
1375O	64	63	9	53
5738O	46	47	16	52
0814O	52	52	6	61
8630O	58	57	8	46
8165O	46	47	14	50
7042O	58	57	5	59
9400O	60	59	7	44
9419N	55	54	14	43
2791N	40	43	16	59
0384N	48	49	12	56
3632N	49	49	6	48
7291N	47	48	6	48
1286N	46	47	8	50
7118N	54	53	17	63
7015N	62	62	12	56
0511O	57	56	5	44
0729O	56	55	20	67
3792O	43	45	11	55
8263O	62	62	7	48
1282N	61	60	11	55

Table B.7: T-Scores of the flow Questionnaire

From the T-scores we can see that about 2/3 of the participants (20) experienced a little bit of flow and on average, we have a T-Score of 53.62 which suggest participants indeed did experience some form of Flow. However our Worry score is also high, with 18 out 29 (62%) showing they experienced some form of anxiety (i.e. afraid of making mistakes, feeling that something important is at stake) while playing with the FC. Looking at the average T-Score of 53.79, suggests that the game will induce these anxiety feelings with participants. Because we are interested in the user's experienced flow while playing with the FC and not very concerned with which version they played on the FC and we are only interested in seeing if they experience more flow while playing with the FC, we can perform a one-tailed one-sampled T-Test with the following H_0 and H_A:
H_0: *Users don't experience Flow while playing the game on the Futuro Cube*
H_A: *Users experience Flow while playing the game on the Futuro Cube*
For Worry, We want to test the same as Flow, we want to detect if Worry is detected while playing the FC, where the version doesn't matter too much. Also because we only care if the user experiences more worry while playing the FC, we only perform a one-tailed one-sampled T-Test with the following H_0 and H_A:
H_0: *Users don't experience Worry while playing the game on the Futuro Cube*
H_A: *Users experience Worry while playing the game on the Futuro Cube*

The Mean of the T-score of Flow is 53.62 and our standard deviation is $\sqrt{\frac{896.84}{28}} \approx 5.65$. Using the one-sample t-test, assuming that the average mean is 50, results in a t score of: $\frac{53.62-50}{5.65\sqrt{29}} \approx 0.11$. This means that our p-value is > 0.1 and we therefore don't experience flow with the game.

The mean of the T-score of Worry is 53.79 and our stand deviation is $\sqrt{\frac{1222.76}{28}} \approx 6.61$. Using the one-sample t-test, assuming that the average mean is 50, results in a t score of: $\frac{53.79-50}{6.61\sqrt{29}} \approx 0.106$. This results in a p-value of > 0.1, which means we don't have a significant difference to suggest users will experience Worry while playing.

B.3.3.2 Performed better

If we were to separate the new versions from the old versions we see that for the old version, 11 out of 14 participants experienced flow, and 8 out of the 14 experienced anxiety/risk. In the old version we have an average T-score of 54.7 for Flow and an average T-score of 53.07 for Worry.
For the new version, we see that 9 out of the 15 participants experienced flow, and 10 out of the 15 experience anxiety/risk. In the new version, we have an average T-score of 52.6 for flow and an average T-score of 54.33 for Worry.
While both versions show that it could lead to both flow and anxiety, we see that the old version scores better by slightly reducing the chance of anxiety and causing more people to experience flow while playing it.
If we look at the statistics, we have that, for the old version we're looking at flow, we have a mean T-score of 54.7 and a standard deviation of $\sqrt{\frac{398.86}{13}} \approx 5.54$. Using the one-sample t-test, assuming that the average mean is 50, results in a t-score of $\frac{54.7-50}{5.34\sqrt{14}} \approx 0.235$ and a degree of freedom of 13. This results in a p-value of > 0.1, which means we don't have a significant difference to suggest users will experience Flow while playing the old version.
If we look at risk in the old version, we have a mean T-score of 53.07 and a standard deviation of $\sqrt{\frac{692.93}{13}} \approx 7.3$. Using the one-sample t-test, assuming that the average mean is 50, results in a t-score of $\frac{53.07-50}{7.3\sqrt{14}} \approx 0.112$ and a degree of freedom of 13. This results in a p-value of > 0.1, which means we don't have a significant difference to suggest users will experience Worry while playing the old version.
If we look at flow in the new version, we have a mean T-score of 52.6 and a standard deviation of $\sqrt{\frac{465.6}{14}} \approx 5.76$. Using the one-sample t-test, assuming that the average mean is 50, results in a t-score of $\frac{52.6-50}{5.76\sqrt{15}} \approx 0.103$ and a degree of freedom of 14. This results in a p-value of > 0.1, which means we don't have a significant difference to suggest users will experience Flow while playing the new version.
If we look at risk in the new version, we have a mean T-score of 54.3 and a standard deviation of $\sqrt{\frac{550,84}{14}} \approx 6.27$. Using the one-sample t-test, assuming that the average mean is 50, results in a t-score of $\frac{54.3-50}{6.27\sqrt{15}} \approx 0.177$ and a degree of freedom of 14. This results in a p-value of > 0.1, which means we don't have a significant difference to suggest users will experience Worry while playing the new version.

Appendix: T- Norms for Flow Total Score and Worry

Tabelle: T-Normen für die Flow-Kurzskala FKS (Flow-Gesamtwert und Besorgnis)

Flow-Gesamtwert					
Rohwert	T	Rohwert	T	Rohwert	T
10	21	32	38	52	52
11	23	33	38	53	52
12	24	34	39	54	53
13	24	35	40	55	54
14	25	36	40	56	55
16	26	37	41	57	56
18	27	38	42	58	57
19	27	39	43	59	58
20	28	40	43	60	59
21	29	41	44	61	60
22	30	42	45	62	62
23	31	43	45	63	63
24	31	44	46	64	64
25	32	45	47	65	65
26	33	46	47	66	67
27	34	47	48	67	68
28	35	48	49	68	69
29	36	49	49	69	70
30	36	50	50	70	74
31	37	51	51		
N=2937					

Besorgnis-Werte					
Rohwert	T	Rohwert	T	Rohwert	T
3	37	10	53	17	63
4	42	11	55	18	64
5	44	12	56	19	66
6	46	13	57	20	67
7	48	14	59	21	70
8	50	15	60		
9	52	16	61		
N=1577					

Figure B.21: The T-Norm table provided by Rheinberg et al. [33]

C. Questionnaires

C.1 Questionnaire Experts - Part I

To what degree does Find the Dot minigame help with training your fine motor skills?

Not at all O O O O O Very helpfull

Which aspects of Find The Dot helped you with training your fine motor skills? (in case of nothing: leave this question open)

(open question)

Are there any improvements you would like to see in the next version of Find The Dot?

(open question)

To what degree does Tap The Side minigame help with training your fine motor skills?

Not at all O O O O O Very helpfull

Which aspects of Tap The Side helped you with training your fine motor skills? (in case of nothing: leave this question open)

(open question)

Are there any improvements you would like to see in the next version of Tap The Side?

(open question)

To what degree does the Shake minigame help with training your fine motor skills?

Not at all O O O O O Very helpfull

Which aspects of Shake helped you with training your fine motor skills? (in case of nothing: leave this question open)

(open question)

Are there any improvements you would like to see in the next version of Shake?

(open question)

To what degree does the Spinning minigame help with training your fine motor skills?

Not at all O O O O O Very helpfull

Which aspects of Spinning helped you with training your fine motor skills? (in case of nothing: leave this question open)

(open question)

Are there any improvements you would like to see in the next version of Spinning?

(open question)

To what degree does RoadRunner help with training your fine motor skills?

Not at all O O O O O Very helpfull

Which aspects of RoadRunner helped you with training your fine motor skills? (in case of nothing: leave this question open)

(open question)

Are there any improvements you would like to see in the next version of RoadRunner?

(open question)

Which minigame would you advise to not change in the next version?

☐ Find The Dot
☐ Tap The Side
☐ Shake
☐ Spinning
☐ RoadRunner
☐ None of the above

Which minigame would you advise to change or improve?

☐ Find The Dot
☐ Tap The Side
☐ Shake
☐ Spinning
☐ RoadRunner
☐ None of the above

Are there any motions you would like to see made in the next version of the game you played? If so, describe precisely what this motion is.
If you think you have a fun idea for a minigame with this motion, you're free to describe it down here as well.

(open question)

C.2 Questionnaire Experts - Part II

Which minigames did you experience a difference from the last time you played it?

☐ Find The Dot
☐ Tap The Side
☐ Shake
☐ Spinning
☐ RoadRunner

To what degree does the new version of Find The Dot minigame help with training your fine motor skills?

Not at all O O O O O Very helpfull

To what degree does the new version of Tap The Side minigame help with training your fine motor skills?

Not at all O O O O O Very helpfull

To what degree does the new version of Shake help with training your fine motor skills?

Not at all O O O O O Very helpfull

To what degree does the new version of Spinning help with training your fine motor skills?

Not at all O O O O O Very helpfull

To what degree does the new version of RoadRunner help with training your fine motor skills?

Not at all O O O O O Very helpfull

To what degree does the new Flip minigame help with training your fine motor skills?

Not at all O O O O O Very helpfull

To what degree does the new Snake minigame help with training your fine motor skills?

Not at all O O O O O Very helpfull

How frustrated were you while playing the game?

Not at all O O O O O O O O O O Extremely

How bored were you while playing the game?

Not at all O O O O O O O O O O Extremely

How challenging were the minigames to complete?

Not at all O O O O O O O O O O Extremely

Which improvement did you experience as a good addition to the new Futuro Cube? (In case of nothing, leave this question open)

(open question)

Are there things you would advise to improve further with the current Futuro Cube? (In case of nothing, leave this question open)

(open question)

Any remaining comments you would like to say about the new version of the Futuro Cube?

(open question)

C.3 Questionnaire Students/Alumni

This questionnaire was one questionnaire where the participants needed to fill in the questions at different parts of the experiment. To represent this, we split this section into two subsections, where the first subsection was given at the start of the experiment and the second subsection was given after the participant had played the game on the Futuro Cube.

C.3.1 Experiment number and Personal questions

Fill here your experiment number. This is done to ensure your data stays anonymized and cannot be linked to you as an individual. (The experiment number will be handed out by the researcher. Ask the researcher for your experiment number in case he hasn't told you yet)

(open question)

Are you...

O A man
O A woman
O I prefer not to say
O Otherwise: <open section for the participant to fill in their gender>

How old are you?
O Between 18 and 21
O Between 21 and 25
O Between 25 and 30
O Older than 30

Do you have physical complaints that reduce your fine motor skills?
O Yes
O Yes, but I don't notice it too much/it doesn't affect me much.
O No

Do you have family members that have problems with their motor skills (both gross motor skills and fine motor skills)?
O Yes
O No
O I prefer not to say

Do you exercise a lot?
O I barely come outside
O I walk outside once in a while
O I exercise regularly
O I sport often

What are your hobbies (gaming, sports, etc.)? If you think you don't have any hobbies, you can skip this question

(open question)

C.3.2 Flow Questions

These questions are a set of 13 statements in which you have to state to what degree you agree or disagree with the statement of your experience with the game of the Futuro Cube.

I feel Just the right amount of challenge.

Not at all O O O O O O O Very much

My thoughts/activities run fluidly and smoothly.

Not at all O O O O O O O Very much

I don't notice time passing.

Not at all O O O O O O O Very much

I have no difficulty concentrating.

Not at all O O O O O O O Very much

My mind is completely clear.

Not at all O O O O O O O Very much

I am totally absorbed in what I am doing.

Not at all O O O O O O O Very much

The right thoughts/movements occur of their own accord.

Not at all O O O O O O O Very much

I know what I have to do each step of the way.

Not at all O O O O O O O Very much

I feel that I have everything under control.

Not at all O O O O O O O Very much

I am completely lost in thought.

Not at all O O O O O O O Very much

Something important to me is at stake here.

Not at all O O O O O O O Very much

I won't make any mistake here.

Not at all O O O O O O O Very much

I am worried about failing.

Not at all O O O O O O O Very much

D. TimeSchedule

Week Number	Rough Date	Goal
17-21	25 April - 20 May	recruit Experts
15-21	25 April - 26 May (at the latest)	Make a playable demo for the experts: - adapt the Futuro Cube so that you can easily make changes to the minigames - be able to actively connect through Bluetooth. - have determined what data should be collected for the experiment
15-24	25 April - 8 June	Gain Experience collecting and processing data.
20-21	6 June - 19 June	Let the experts play with the Futuro Cube (collect data).
20-26	6 June - 14 July (at the latest)	Adapt the gameplay: - Add the DDA system. - Add new minigames if the experts want new motions added to the minigame series. - Improve some of the minigames if the experts suggest so.
27-28	15 July - 26 July	follow-up experiment with the experts (collect data).
28-31	15 July - 31 July	Analyze the collected data.
31-32	24 July - 5 August	write Results and conclusion
32-33	1 August - 16 August	prepare Thesis defense
33-34	12 August - 16 August	Thesis defense and finishing touches thesis.

Table D.1: Planned Time Schedule

Bibliography

[1] R. F. Cube, *Rubik's futuro cube*. [Online]. Available: https://www.futurocube.com/.

[2] A. Brons, A. de Schipper, S. Mironcika, *et al.*, "Assessing children's fine motor skills with sensor-augmented toys: Machine learning approach," *Journal of Medical Internet Research*, vol. 23, no. 4, e24237, 2021.

[3] R. H. Geuze, M. J. Jongmans, M. M. Schoemaker, and B. C. Smits-Engelsman, "Clinical and research diagnostic criteria for developmental coordination disorder: A review and discussion," *Human movement science*, vol. 20, no. 1-2, pp. 7–47, 2001.

[4] P. H. Wilson, S. Ruddock, B. Smits-Engelsman, H. Polatajko, and R. Blank, "Understanding performance deficits in developmental coordination disorder: A meta-analysis of recent research," *Developmental Medicine & Child Neurology*, vol. 55, no. 3, pp. 217–228, 2013.

[5] L. M. S. Dylan P Cliff Antony D Okely and K. McKeen, "Relationships between fundamental movement skills and objectively measured physical activity in preschool children," *Pediatric exercise science*, 2009.

[6] S. C. Cools W Martelaer KD and A. C., "Movement skill assessment of typically developing preschool children: A review of seven movement skill assessment tools," *J Sports Sci Med*, 2009.

[7] A. S. Grissmer D Grimm KJ, "Fine motor skills and early comprehension of the world: Two new school readiness indicators," *Dev Psychol*, 2010.

[8] B. S.-E. S.E. Henderson D.A. Sugden, *Movement abc-2 nl | movement assessment battery for children*. [Online]. Available: https://www.pearsonclinical.nl/movement-abc-2-nl-movement-assessment-battery-children.

[9] K. F. Strooband, M. de Rosnay, and A. D. Okely, "Prevalence and risk factors of preschoolers' fine motor delay within vulnerable australian communities," *Journal of Paediatrics and Child Health*, vol. 57, no. 1, pp. 114–120, 2021.

[10] A. B. S.e. Henderson, *Movement assesment battery for children - third edition*. [Online]. Available: https://www.pearsonassessments.com/store/usassessments/en/Store/Professional-Assessments/Motor-Sensory/Movement-Assessment-Battery-for-Children%2C-Third-Edition/p/P100051002.html.

[11] D. Rivera, A. García, B. Alarcos, J. R. Velasco, J. E. Ortega, and I. Martínez-Yelmo, "Smart toys designed for detecting developmental delays," *Sensors*, vol. 16, no. 11, p. 1953, 2016.

[12] S. Mironcika, A. de Schipper, A. Brons, H. Toussaint, B. Kröse, and B. Schouten, "Smart toys design opportunities for measuring children's fine motor skills development," in *Proceedings of the Twelfth International Conference on Tangible, Embedded, and Embodied Interaction*, 2018, pp. 349–356.

[13] J. Sander, A. de Schipper, A. Brons, *et al.*, "Detecting delays in motor skill development of children through data analysis of a smart play device," in *Proceedings of the 11th EAI International Conference on Pervasive Computing Technologies for Healthcare*, 2017, pp. 88–91.

[14] R. F. Cube, *Rubik's futuro cube suite*. [Online]. Available: https://www.futurocube.com/support/.

[15] R. F. Cube, *Code.futurocube.com*. [Online]. Available: https://code.futurocube.com/.

[16] M. Csikszentmihalyi, "The flow experience and its significance for human psychology," in *Optimal Experience: Psychological Studies of Flow in Consciousness*, M. Csikszentmihalyi and I. S. Csikszentmihalyi, Eds. Cambridge University Press, 1988, pp. 15–35.

[17] J. Chen, "Flow in games (and everything else)," *Communications of the ACM*, vol. 50, no. 4, pp. 31–34, 2007.

[18] M. Zohaib *et al.*, "Dynamic difficulty adjustment (dda) in computer games: A review," *Advances in Human-Computer Interaction*, vol. 2018, 2018.

[19] G. Andrade, G. Ramalho, H. Santana, and V. Corruble, "Extending reinforcement learning to provide dynamic game balancing," in *Proceedings of the Workshop on Reasoning, Rep-*

resentation, and Learning in Computer Games, 19th International Joint Conference on Artificial Intelligence (IJCAI), 2005, pp. 7–12.

[20] S. Xue, M. Wu, J. Kolen, N. Aghdaie, and K. A. Zaman, "Dynamic difficulty adjustment for maximized engagement in digital games,"

[21] C. Pedersen, J. Togelius, and G. N. Yannakakis, "Modeling player experience in super mario bros," in *2009 IEEE Symposium on Computational Intelligence and Games*, IEEE, 2009, pp. 132–139.

[22] P. Spronck, I. Sprinkhuizen-Kuyper, and E. Postma, "Online adaptation of game opponent ai with dynamic scripting," *International Journal of Intelligent Games and Simulation*, vol. 3, no. 1, pp. 45–53, 2004.

[23] R. Hunicke and V. Chapman, "Ai for dynamic difficulty adjustment in games," *Challenges in game artificial intelligence AAAI workshop*, vol. 2, Jan. 2004.

[24] J. Hagelback and S. J. Johansson, "Measuring player experience on runtime dynamic difficulty scaling in an rts game," in *2009 IEEE Symposium on Computational Intelligence and Games*, IEEE, 2009, pp. 46–52.

[25] X. Li, S. He, Y. Dong, *et al.*, "To create dda by the approach of ann from uct-created data," in *2010 international Conference on computer application and system modeling (ICCASM 2010)*, IEEE, vol. 8, 2010, pp. V8–475.

[26] A. Stein, Y. Yotam, R. Puzis, G. Shani, and M. Taieb-Maimon, "Eeg-triggered dynamic difficulty adjustment for multiplayer games," *Entertainment computing*, vol. 25, pp. 14–25, 2018.

[27] D. Afergan, E. M. Peck, E. T. Solovey, *et al.*, "Dynamic difficulty using brain metrics of workload," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2014, pp. 3797–3806.

[28] H. D. Fernandez B., K. Mikami, and K. Kondo, "Adaptable game experience based on player's performance and eeg," in *2017 Nicograph International (NicoInt)*, 2017, pp. 1–8. DOI: 10.1109/NICOInt.2017.11.

[29] A. Ebrahimi and M.-R. Akbarzadeh-T, "Dynamic difficulty adjustment in games by using an interactive self-organizing architecture," in *2014 Iranian Conference on Intelligent Systems (ICIS)*, IEEE, 2014, pp. 1–6.

[30] H. Avilés, R. Luis, J. Oropeza, *et al.*, "Gesture therapy 2.0: Adapting the rehabilitation therapy to the patient progress," *Probabilistic Problem Solving in BioMedicine*, p. 3, 2011.

[31] M. Pirovano, R. Mainetti, G. Baud-Bovy, P. L. Lanzi, and N. A. Borghese, "Self-adaptive games for rehabilitation at home," in *2012 IEEE Conference on Computational Intelligence and Games (CIG)*, IEEE, 2012, pp. 179–186.

[32] N. Semiconductor, *Nrf connect android app*. [Online]. Available: https://play.google.com/store/apps/details?id=no.nordicsemi.android.mcp&hl=nl&pli=1.

[33] F. Rheinberg, R. Vollmeyer, and S. Engeser, "Flow short scale," *Motivation and Emotion*, 2003.

[34] ElevenLabs, *Elevenlabs*. [Online]. Available: https://elevenlabs.io/.

[35] A. Team, *Audacity*. [Online]. Available: https://www.audacityteam.org/.

[36] R. F. Cube, *Game library futuro cube*. [Online]. Available: https://www.futurocube.com/games/.

[37] P. Thijssens. "Motor skill development game - futuro cube," Youtube. (2021), [Online]. Available: https://www.youtube.com/watch?v=XYqxu5C8x3E.

[38] Kellor, Frost, SilberBerg, Iversen, and Cummings, *Nine peg hole test*. [Online]. Available: https://www.physio-pedia.com/Nine-Hole_Peg_Test.

[39] F. H. William Sealy Gosset and J. Lüroth, *T table*. [Online]. Available: https://www.tdistributiontable.com/.