Chair for Computer Science 10 (Media Computing and Human-Computer Interaction)

RWTH AACHEN UNIVERSITY

Comparison of Controller-Based vs. Touch-Based Input on Large Displays

Bachelor's Thesis at the
Media Computing Group
Prof. Dr. Jan Borchers
Computer Science Department
RWTH Aachen University

by
Nasim Khatib

Thesis advisor:
Dr. Simon Völker

Second examiner:
Prof. Dr. Ulrik Schroeder

Registration date: 17.10.2024
Submission date: 18.03.2025

Zentrales Prüfungsamt/Central Examination Office

Eidesstattliche Versicherung
Declaration of Academic Integrity

Khatib, Nasim

Name, Vorname/Last Name, First Name

375512

Matrikelnummer (freiwillige Angabe)
Student ID Number (optional)

Ich versichere hiermit an Eides Statt, dass ich die vorliegende Arbeit/Bachelorarbeit/ Masterarbeit* mit dem Titel
I hereby declare under penalty of perjury that I have completed the present paper/bachelor's thesis/master's thesis* entitled

Comparison of Controller-Based vs. Touch-Based Input on Large Displays

selbstständig und ohne unzulässige fremde Hilfe (insbes. akademisches Ghostwriting) erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt; dies umfasst insbesondere auch Software und Dienste zur Sprach-, Text- und Medienproduktion. Ich erkläre, dass für den Fall, dass die Arbeit in unterschiedlichen Formen eingereicht wird (z.B. elektronisch, gedruckt, geplottet, auf einem Datenträger) alle eingereichten Versionen vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.
independently and without unauthorized assistance from third parties (in particular academic ghostwriting. I have not used any other sources or aids than those indicated; this includes in particular software and services for language, text, and media production. In the event that the work is submitted in different formats (e.g. electronically, printed, plotted, on a data carrier), I declare that all the submitted versions are fully identical. I have not previously submitted this work, either in the same or a similar form to an examination body.

Aachen, 01.02.2025

Ort, Datum/City, Date

Unterschrift/Signature

*Nichtzutreffendes bitte streichen/Please delete as appropriate

Belehrung:
Official Notification:

§ 156 StGB: Falsche Versicherung an Eides Statt
Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.
§ 156 StGB (German Criminal Code): False Unsworn Declarations
Whosoever before a public authority competent to administer unsworn declarations (including Declarations of Academic Integrity) falsely submits such a declaration or falsely testifies while referring to such a declaration shall be liable to imprisonment for a term not exceeding three years or to a fine.
§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt
(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.
(2) Straflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.
§ 161 StGB (German Criminal Code): False Unsworn Declarations Due to Negligence
(1) If an individual commits one of the offenses listed in §§ 154 to 156 due to negligence, they are liable to imprisonment for a term not exceeding one year or to a fine.
(2) The offender shall be exempt from liability if they correct their false testimony in time. The provisions of § 158 (2) and (3) shall apply accordingly.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:
I have read and understood the above official notification:

Aachen, 01.02.2025

Ort, Datum/City, Date

Unterschrift/Signature

Contents

List of Figures and Tables

Abstract

The growing usage of large displays in our daily lives has led us to consider the best and most suitable input methods to interact with such displays. Although touch has become a common interaction method used with our mobile devices, it can be challenging to apply it to longer periods of time with large displays. This challenge could be solved by replacing the touch interaction method with the common console controller.
For this thesis, a video game was created with the intention of exploring and comparing the difference between touch-based and controller-based input methods. In addition, a user study was conducted in which the developed game was tested individually on 11 participants.
The video game consists of 10 levels with three stages of difficulty, requiring the users to coordinate their hand movements with their cognitive abilities. This thesis therefore outlines the process and results of the game creation and the user study, thereby providing a comparison between the two interaction methods.

Überblick

Die zunehmende Nutzung großer Bildschirme in unserem Alltag hat dazu geführt, dass wir die besten und bestgeeigneten Eingabemethoden für die Interaktion mit solchen Displays in Betracht ziehen. Obwohl die Touch-Eingabe zu einer gängigen Interaktionsmethode auf unseren mobilen Geräten geworden ist, kann ihre Anwendung über längere Zeiträume auf großen Bildschirmen herausfordernd sein. Diese Herausforderung könnte durch den Ersatz der Touch-Interaktion mit einem herkömmlichen Konsolen-Controller gelöst werden.
Für diese Arbeit wurde ein Videospiel entwickelt, um die Unterschiede zwischen touch-basierten und controllerbasierten Eingabemethoden zu erforschen und zu vergleichen. Zudem wurde eine Nutzerstudie durchgeführt, in der das entwickelte Spiel von 11 Teilnehmenden individuell getestet wurde.
Das Videospiel besteht aus 10 Levels mit drei Schwierigkeitsstufen und erfordert von den Nutzern die Koordination ihrer Handbewegungen mit ihren kognitiven Fähigkeiten. Diese Arbeit beschreibt daher den Entwicklungsprozess und die Ergebnisse des Spiels sowie der Nutzerstudie und liefert einen Vergleich zwischen den beiden Interaktionsmethoden.

Acknowledgments

I would like to thank my supervisor, for being very supportive and reachable when needed, and great thanks to the unity tutorials available on the web, that have proven to be crucial and very helpful in times of confusion, and for the wide and precise documentation of unity to it's functions on its website. Furthermore, I would like to thank everyone that has supported me throughout the thesis, and many thanks to the participants that took part in the study, who have given very positive critical and very informative feedback in the user study.

Chapter 1

Introduction

1.1 Motivation

Interactions with electrical devices such as phones, laptops and TVs have become almost mandatory for our daily life, where almost everything is being processed through devices, e.g. instead of calling the doctor's office, one can look for appointment options on the internet, instead of buying multiple books and having to carry them around, one can buy an e-book, and have multiple books on it ready to read and easy to travel with, instead of writing notes and articles on paper, one can use a digital notepad.
Interacting with the digital world requires an important component, displays, where all the process is projected on. We have noticed that the size of the displays has a significant influence on the interaction methods used to engage with the devices. For instance, smartphones and tablets, having relatively small displays, predominantly rely on touch-based input methods, offering gestures that have become intuitive to us over years of experience and daily utilization, such as tapping, swiping, pinching, stretching, drag and dropping. In another instance, computers, which are typically featured through larger screens, offer a combination between a keyboard and a mouse, where the movement of the mouse navigates a cursor around the display, and the keyboard for giving commands and typing.

As displays increase in size, such as those found in conference rooms, lecture halls, and offices, alternative interaction methods become necessary.
In our paper, we are going to focus on 2 methods and compare them, which consist of touch-based and controller-based (console controller) input methods, when used with large displays.

1.2 Process of the Thesis work

In order to be able to compare the two input methods (touch and controller), we had to figure out a task to use with the input methods and a large display that has touch sensors.
After some thinking, we came to the conclusion that a video game is a good candidate, since video games can vary in tasks and requirements, and usually offer a challenge to motivate the user, which can make the user enjoy their participation in the study and shift their focus away from the fact that they are undertaking a study, which makes the process of the study more comfortable for the user and more authentic.
Our next step was to brainstorm game ideas that can offer a challenge, through which it is possible to compare the utilization of controller-inputs and touch-inputs, and we decided to develop a puzzle game based on a board game called "Geoboard", as shown in Figure 3.1. The goal of the game is for the players to form certain shapes using stretchy objects.
When we finished developing the game, we started the process of user testing, where we asked 11 participants to finish a combined total of 10 levels, 5 of which with touch-based input and 5 with controller-based input, and we calculated the amount of steps they required to complete each level, the duration of each level, which was limited to 5 minutes for each try, although given the option to restart the level and therefore the timer as well, and whether they were able to finish the level or not! Afterwards we analyzed the outputs and the opinions of each user.

1.3 Thesis Goals

The goal of this thesis is to research possible compatible methods to interact with large displays, were we compare 2 common input methods across a video game that tests the effect of the input method on the time required to complete the levels of the video game and the amount of moves needed. We also asked the participants for some feed back through a user evaluation. Through analyzing the data from both the feedback and the results from the video game, we will be able to conclude the best method out of the two, for utilization with large displays.
Another goal is to open the possibilities for future studies regarding the utilization of large displays in our daily life, with different input methods as well.

Chapter 2

Related Work

2.1 Touch-Based Input

Due to the proliferation of touch screens, touch-based input methods such as drag-and-drop, pinching, swiping, and clicking or selecting, became so intuitive that we utilize them unwittingly, even so that we see almost every other individual walking and using their mobile screen. The paper of MacKay et al. [2005] investigates similar touch input methods on mobile devices in different environments that we go through everyday (sitting, standing and walking), indicating that different environments require different input methods to maximize efficiency and comfort, even between touch-based input methods themselves. Other papers also study the efficiency of using both hands or just one hand, which proved that, it is not necessarily optimal to use two handed input when interacting with touch screens, when interacting with complex tasks using two handed input methods may complicate the task even further, by forcing coordination between the hands, as shown in Kabbash et al. [1994]. On the other hand Owen et al. [2005] argue that bimanual (two handed) interactions offer a significant advantages in tasks that require intricate manipulation, suggesting that utilizing two-handed techniques can enhance user experience and efficiency in what the authors called " curve manipulation".
Touch-based input devices are used daily, but how do they

function and how is the touch registered onto the device. The paper Hoye and Kozak [2010] focuses on three primary touch screen technologies (resistive, capacitive, and infrared), examining the evolution, application, and sustainability of those technologies, and comparing them with each another, the authors also highlight the importance of selecting appropriate touch screen systems based on specific application requirements.

2.2 Controller-Based Input

The evolution of video game controllers is remarkable and has been going for longer than one would expect, starting as simple as an arcade joystick and button, until controllers as we know nowadays. The author Lu [2008] has given us a comprehensive view of this evolution, highlighting the crucial role it played in shaping the interactive nature of modern video games, reflecting the industry's continuous improvement of user engagement and control precision it offers with every step.
Another study worth mentioning is the case study of Young et al. [2016], which compares different types of controllers with each other using simple two-dimensional pointing tasks, aiming to evaluate user satisfaction and usability.
While the previous paper focuses on comparing user experience with different controllers, the authors of Brown and MacKenzie [2013] research the relation of the usability of the common console controller with users of different hand size. In their findings they confirm that the hand size of users significantly impacted usability, and the overall experience with the video games.
The previous papers highlight the vast research opportunities that controllers present, which demonstrates the need for further exploration into their design, usability, and adaptability to diverse user needs.

2.3 Controller vs Touch

In their paper Zaman et al. [2010] compare touch-based and controller-based input methods on a small screen using an Apple iPhone for touch-based input and the Nintendo DS for controller-based input. This comparison was utilized through a video game available on both devices, named "Assassin's Creed: Altair's Chronicles" and developed by Ubisoft[1]. The author's findings suggest that when interacting with smaller displays it might be more beneficial to use controller-based input methods, a concrete reason being, that touch-based inputs have crucial disadvantages (e.g. "the finger occludes parts of the display, covering up valuable pixels and making it harder to see the results of interactions" as Butler et al. [2008] suggested).
Using controllers for tasks that requires texting and typing, can be quite challenging, since navigating between the letters with the common joystick would be fairly time consuming, in comparison to touch-based or keyboard-based input methods, which provide a more intuitive and comfortable navigation between letters, offering greater efficiency and precision for text input compared to controller-based alternatives. Their design allows for quicker access and more fluid typing, making them more effective for efficient text entry.
Despite keyboard-based and touch-based input methods clearly offering more efficient text entry, as shown in Boletsis and Kongsvik [2019], the authors Wilson and Agrawala [2006] still investigate the potential of gaming console controllers (using an Xbox gaming console, which is quite identical to the PS5 controller used in our study) as a viable alternative for text input.
Other studies like Boletsis and Kongsvik [2019] that have used different types of controller (virtual reality controllers) also show that despite having better utility, touch-based and keyboard-based text input methods are still more preferable. Therefore, controller compatible keyboards were developed such as the "PS5 Controller

[1] https://www.ubisoft.com/en-us/

Keyboard" (module:TP5-0556) produced by dobe[2]

2.4 Large Displays

Large displays have become very common, we see them on the streets, whether at a bus station for advertisement, in offices, even at home as a television screen, and in malls. With increase in usage and the diversity in which they get used for, offers the academic industry, many research opportunities.
The paper of Ardito et al. [2015] provides a comprehensive overview of the research and developments in the field of human-computer interaction concerning large displays, it examines various interaction techniques, usability challenges, and the implications of using large displays in different contexts.
The authors of Tan et al. [2006] even investigate how the size of the display influences the user's performance specifically on spatial tasks, arguing that larger displays enhance performance on such tasks. In their findings, the authors suggest that incorporating larger displays can significantly benefit tasks requiring spatial awareness and manipulation, potentially informing design decisions in various applications, from data visualization to interactive environments.

2.5 Other common input methods

Although we are mainly focusing on comparing touch-based and controller-based input methods in this thesis, there are other input methods that can be utilized with interactive devices, like the common mouse and keyboard, gesture-based input (Kim and Lee [2016]) which is widely common with augmented reality, virtual reality's handheld controllers, or even speech-based input methods (Pandey et al. [2021]).

[2] https://www.dobe-game.com/en/productshow-90-811.html

All of the previously mentioned input methods fit different tasks best, for example, speech-based input methods work best for texting and typing. Pandey et al. [2021] even investigate silent speech-based input, which focuses on the movement of the tongue and lips to register inputs, or gesture-based input methods, which fits augmented reality or virtual screens that cannot offer touch inputs, but simulate such through gestures of pinching selecting and hand and finger motion tracking, as shown in Paay et al. [2017].

Chapter 3

Own work

In this chapter we are going to talk about the video game that has been developed, which requires the player to drag and drop nodes attached to specific positions of a shape to reform it to match the required shape as shown in the upper-middle part of the screen of each level, the player has to complete 5 levels using touch input, and the other 5 using controller input, to accomplish the game.

3.1 Background of the game

The idea of the game is based on a cognitive board game called "Geoboard" 3.1, which is widely used in elementary schools and pre-schools for educational purposes. Teaching the players the fundamentals of geometry.
The game includes a board (usually wooden) with nails equidistant from one another, and rubber bands.
Using the previously mentioned tools, the player is supposed to form a certain shape on the board, by streching the rubber bands around the nails on the board, as seen in Figure 3.1. The video game that has been developed based on "Geoboard" 3.1, has been called "ShapeShifter", the white nodes on the screen simulate the nails on the actual board game, the shape is supposed to be the rubber band, and the red nodes on the shape represent points of

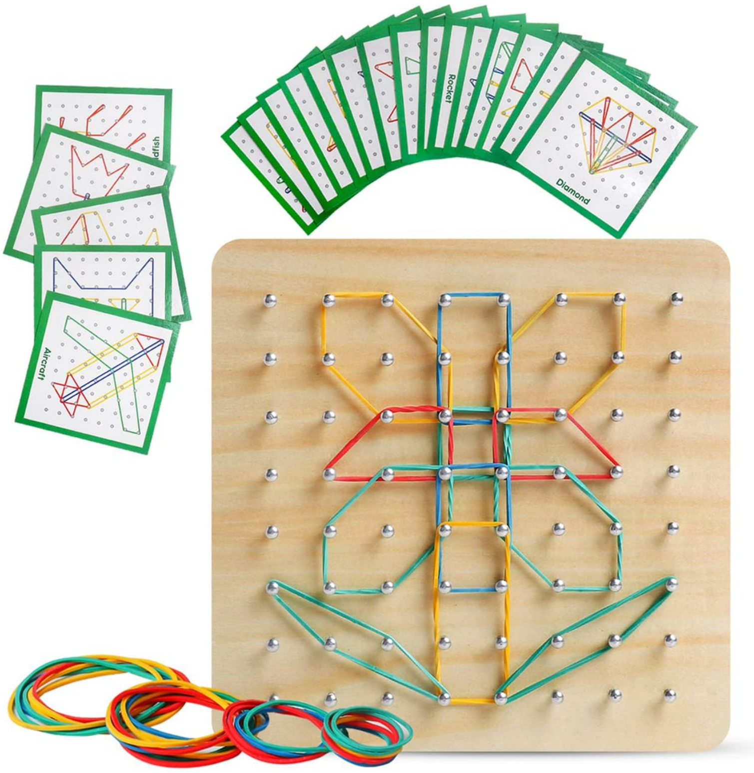


Figure 3.1: Geoboard, is the board-game, which the game that we have developed is based on. Image source: https://www.amazon.com/Wooden-Geoboard-Mathematical-Manipulative-Material/dp/B07PHPLZX5

control, which are the positions on the shape which the player can interact with and drag around to form the required shape.

3.2 Design and Implementation of the Game

The game has been developed using unity[1] (version 2022.3.33f1) as a game engine. Ten levels have been created, divided into 3 difficulties (easy, medium and hard), 4

1 `https://unity.com/`

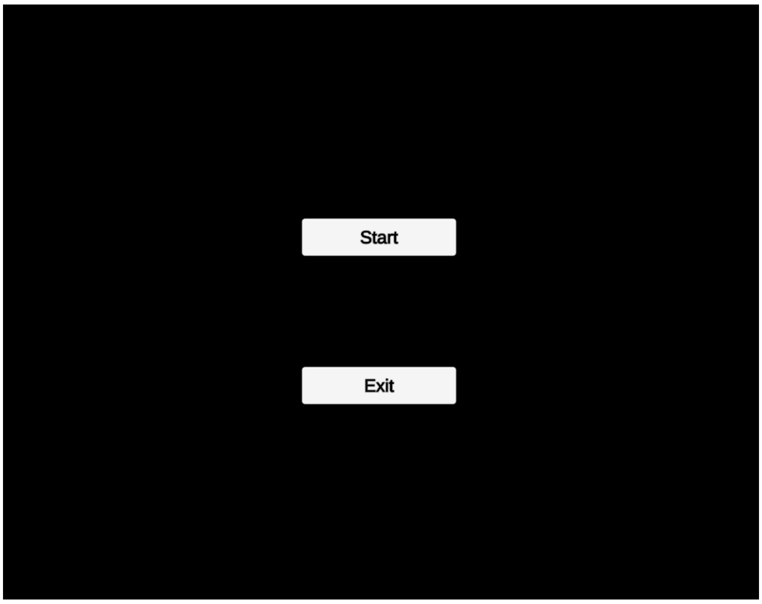


Figure 3.2: Main menu screen

easy levels, 4 medium, and 2 hard levels, so that 5 can be played with each interaction method (2 easy, 2 medium and 1 hard). The distribution of the levels for each difficulty will be mentioned in later chapters.
As seen in Figure 3.2 the game starts with a main menu screen, after pressing the start button, the user will be led to a new screen that asks the user to insert their user-ID as seen in Figure 3.3, each user has been given a user-ID, based on their position in the user study.

The user-ID is important for attaching each outcome of the levels to the user, without using their actual names for anonymity.
The next step will be for the user to choose an interaction method as seen in Figure 3.4, to avoid biased outcomes, we have adviced the participants with an odd user-ID numbers to start with controller and the participants with even User-ID numbers 5 to start with touch.

In Figure 3.5 we can see an example of the layout of each level (taken from the first level). We can observe a small

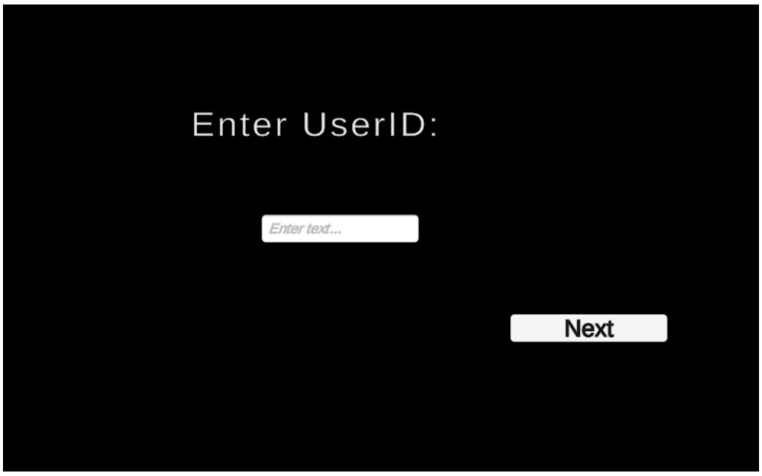


Figure 3.3: UserID screen, that is shown after the user presses the start button

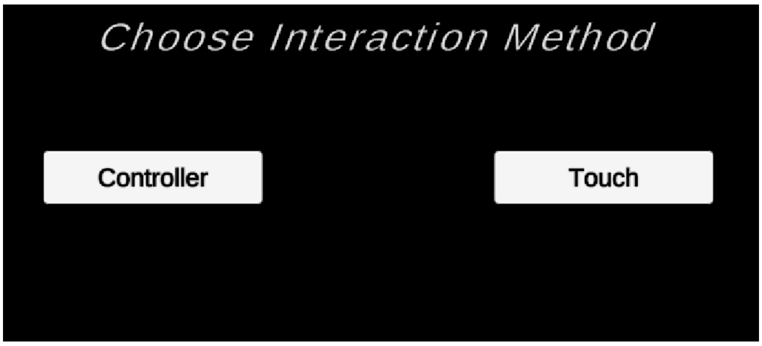


Figure 3.4: The user is given the Choice to use the preferred method to begin with through this screen

difference in the 2 images, as illustrated in Figure 3.5, all the nodes in the upper part of the image are red, while in the lower part of the image, one node is blue. The blue node represents the current position (the currently selected node) in the controller interaction method. This is due to the difference between the interaction strategies of each input method.

The level's layout offers 3 buttons as seen in Figure 3.5:

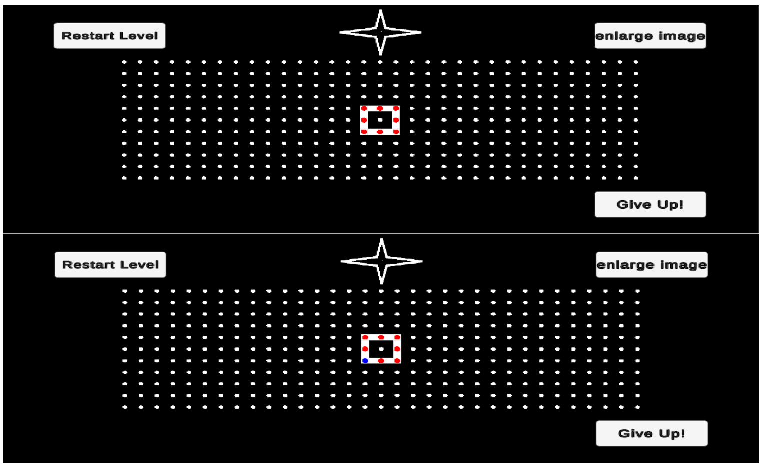


Figure 3.5: the upper part of the image represents Level_1 when applied with the touch interaction method, and the lower part when applied with the controller interaction method

- "Restart Level": Can be found on the top left corner of the screen of each level the player is offered a chance to restart the level, which then restarts the positions of all nodes into their original position and restarts the timer on each try, giving the player the opportunity to finish the level.
- "enlarge Image": Positioned on the top right corner of the screen, offering the player an option to expand the solution image shown in the top middle part of the screen, making it clearer for the player.
- "Give Up!": As can be seen in the lower right corner of the screen, is a button that allows the player to skip the current level if desired or take a small break if needed. When pressed, this button opens a canvas as shown in Figure 3.6, giving the player the choice to resume the level or skip to the next level.

The levels have been categorized into three distinct difficulty groups: easy, medium, and hard. In each level, the player is required to construct specific shapes to successfully complete it:

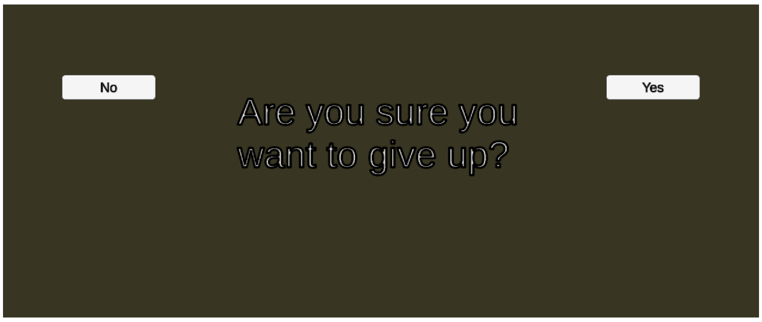


Figure 3.6: Canvas shown after pressing the "Give Up!" button

- **Group 1:** consists of 4 easy levels, with the shapes, as shown in Figure 3.7, given the following Level_IDs:
  - **Level_1**: 4 point star.
  - **Level_2**: Arrow.
  - **Level_3**: House.
  - **Level_4**: Rocket.
- **Group 2:** consists of 4 medium levels, with the shapes, as shown in Figure 3.8, given the following Level_IDs:
  - **Level_5**: Clubs of cards.
  - **Level_6**: Sigma letter.
  - **Level_7**: Sword.
  - **Level_8**: Key.
- **Group 3:** consists of 2 hard levels, with the shapes, as shown in Figure 3.9, given the following Level_IDs:
  - **Level_9**: Christmas tree.
  - **Level_10**: batman symbol.

The distribution of the levels into the difficulty group has been determined by multiple factors, including the number of the nodes, the overall complexity of the shape, and the ranges of the angles of each of the nodes (more on this in Levels Design 3.2.1).

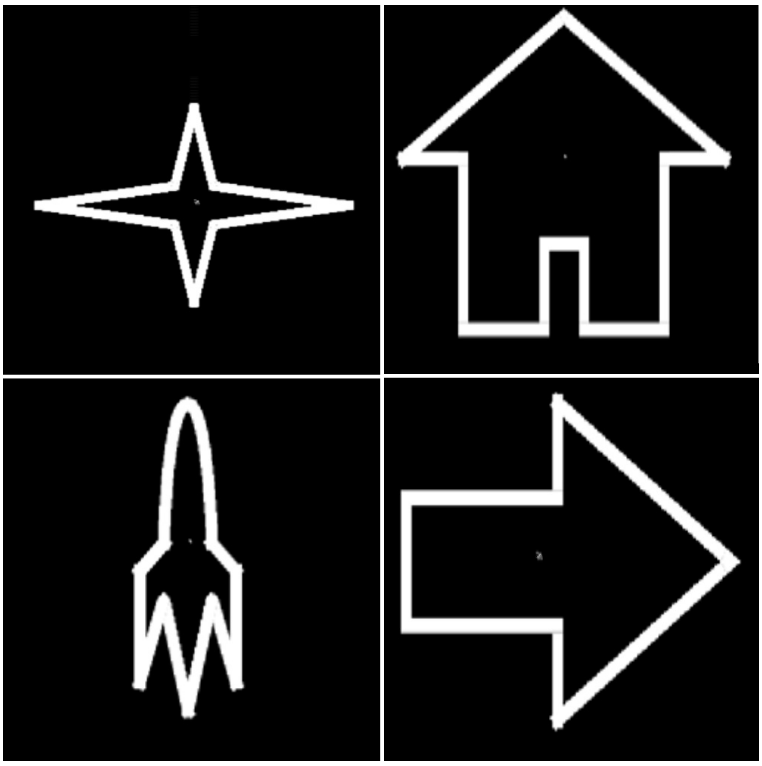


Figure 3.7: Shapes of the medium difficulty level group, from

3.2.1 Levels Design

Each level in the game is characterized by a specific number of interaction points, which define the key locations where the player can manipulate the shapes. The complexity of the level increases with the number of these interaction points(From this point forward, the term 'interaction points' will be referred to as 'nodes'), influencing the possible configurations and the difficulty of achieving the correct shape. The completion criteria for each shape are determined based on predefined geometric constraints, such as the angle thresholds and equality between angles that are symmetrical. The solution of each level is definitive, which means that it doesn't matter, how one rotates the nodes, if the shape is formed correctly, it will be completed. This is due to the first node always being the node that has the

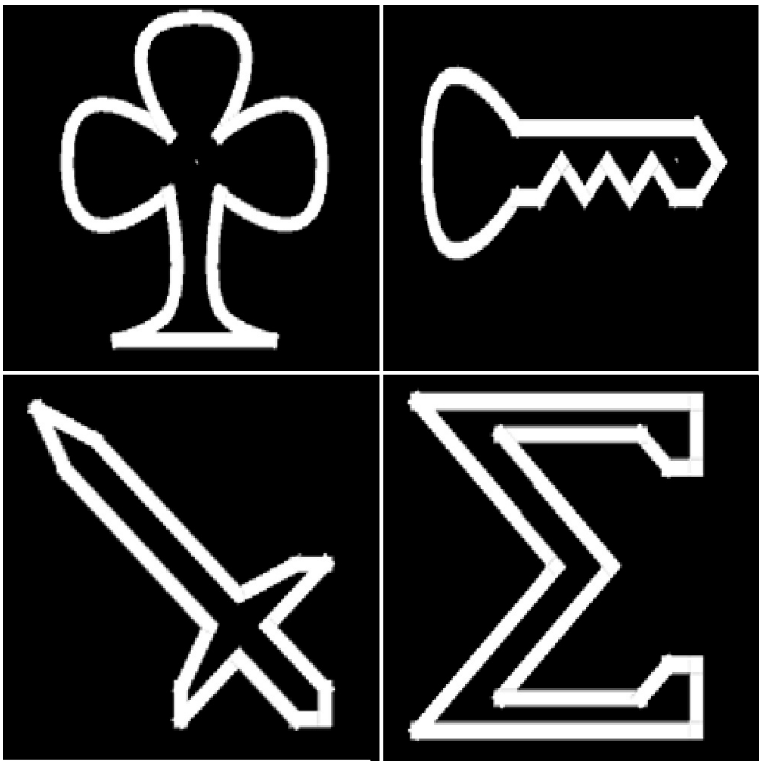


Figure 3.8: Shapes of the medium difficulty level group, from

lowest x position and highest y position between all other nodes.

3.3 Game Functionality and Mechanics

As mentioned in the previous section 3.2.1, each level has a specific amount of nodes and each node has an angle threshold. In this section we are going to talk about the mechanics behind it.
After deciding on using angles and symmetry between nodes as a satisfiability method to compare the shapes to their pattern, we had to start implementing it into code, for that we used vector calculation, where we had to calculate the vectors going from each node to it's 2 neighboring nodes. Following method 3.1 is used to do so:

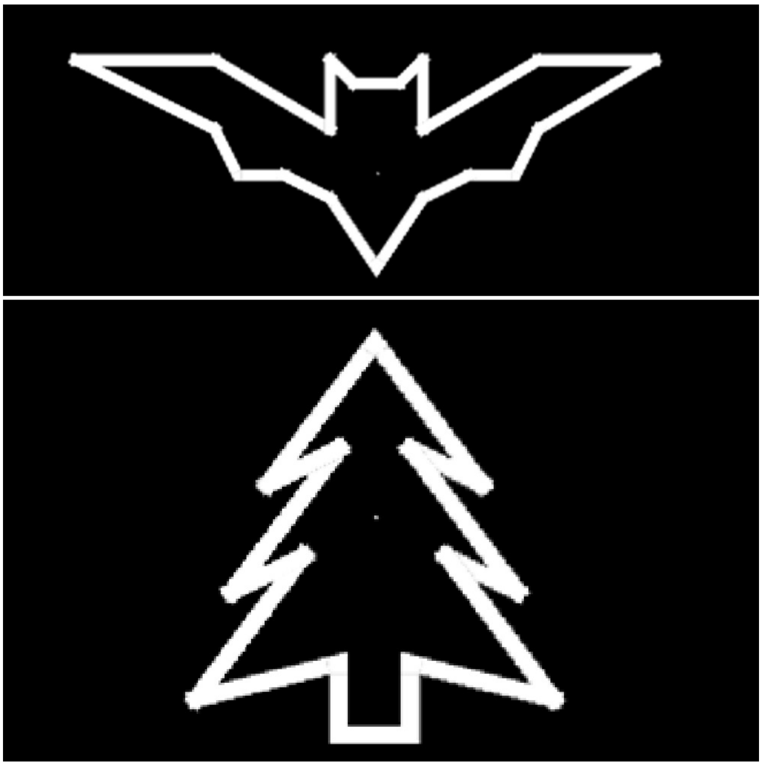


Figure 3.9: Shapes of the medium difficulty level group, from

```
public  float GetAngle(Vector2 position1,
    Vector2 position2, Vector2 position3)
    {
        // Calculate direction vectors
        Vector2 dir1 = position2 - position1;
        Vector2 dir2 = position2 - position3;

        // Calculate the magnitudes of the
           direction vectors
        float mag1 = dir1.magnitude;
        float mag2 = dir2.magnitude;

        // Calculate the dot product
        float dotProduct = Vector2.Dot(dir1,
           dir2);

        // Calculate the angle in radians
        float angleRad = Mathf.Acos(
           dotProduct / (mag1 * mag2));
```

```

        // Convert radians to degrees
        float angleDeg = angleRad * Mathf.
           Rad2Deg;
        //Debug.Log(angleDeg);
        return angleDeg;

    }
```

Listing 3.1: GetAngle Method in C#

Through $GetAngle$ 3.1, we can calculate the angle of each node by assigning the left neighboring node to $position1$, the desired node to $position2$, and the neighboring right node to $position3$, which we have applied in the $CalcAngle$ method.
After calculating the angle of each node, we use the predefined Update Method to track the angle each frame, in order to have a real time measurement throughout the level. Since we now have a real time measurement, we can manually check at what angles the nodes need to be in, in order for the shape to match it's pattern, then make a certain threshold. To ensure the game remains solvable, we need to allow multiple possible solutions. For example, in level_1, any angle within the following range can be used for their respective nodes:

- Node_0: $11.4 < X < 22.7$
- Node_1: $107.5 < X < 115.4$
- Node_2: $22.6 < X < 36.9$
- Node_3: $107.5 < X < 115.4$
- Node_4: $11.4 < X < 22.7$
- Node_5: $107.5 < X < 115.4$
- Node_6: $22.6 < X < 36.9$
- Node_7: $107.5 < X < 115.4$

When introducing a tolerance range to make it easier for players to match the shape to the solution, we needed to ensure that certain angles remained identical to maintain symmetry. This symmetry is essential for the shapes to align correctly. As a result, we implemented an additional method to verify shape matching. This method checks whether specific nodes are symmetrical to others that are visually expected to be symmetrical. For example, in Figure 3.10, the following symmetry between nodes can be observed:

- Node_0 symmetrical to node_12
- Node_1 symmetrical to node_11
- Node_2 symmetrical to node_10
- Node_3 symmetrical to node_9
- Node_4 symmetrical to node_8
- Node_5 symmetrical to node_7

Although nodes 6 and 13 appear symmetrical in the image, we decided not to include them in the symmetry criteria to reduce complexity. During the pre-testing phase, we observed that enforcing this symmetry made the level significantly more challenging.

3.4 Input Methods

In this section we will discuss the input methods used in our study, namely touch-based and controller-based input methods, and elaborate on their respective mechanics.

3.4.1 Touch Input

Touch-based input method has been primarily utilized in a drag-and-drop approach (which has also been tested on large display by the authors Doeweling and Glaubitt [2010]), where the participants have to interact

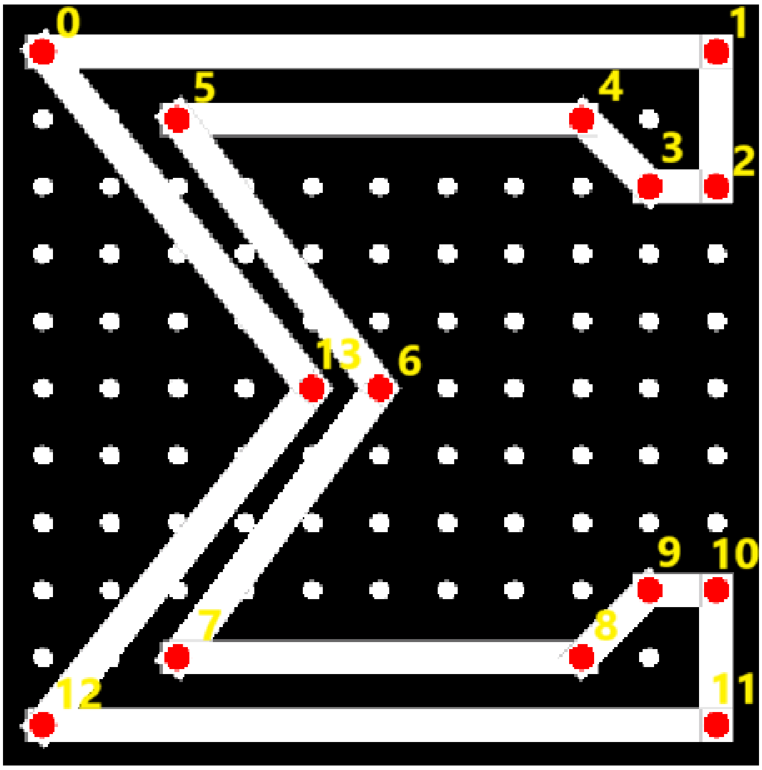


Figure 3.10: starting with the top right node as node 0 then proceeding to the net node clockwise

with the nodes by dragging them to the desired position and then releasing or dropping them, close to that position.

The video game supports only single-touch input methods (meaning only one touch input can be registered at a time), as the interaction method did not require multiple simultaneous touch points.
The video game provides the user with two types of interactions throughout the levels: either by clicking one of the three buttons shown in Figure 3.5, or by dragging and dropping the red nodes in the interactive shape. For the touch-based interactions with the shape, we employed the following method:

```
void TouchHandleInput()
    {
        if (IsPointerOverUIElement())
        {
            return;
        }
        if (Input.GetMouseButtonDown(0))
        {
            Vector3 mousePosition = Camera.main.ScreenToWorldPoint(Input.mousePosition);
            mousePosition.z = 0;
            Collider2D[] colliders = Physics2D.OverlapPointAll(mousePosition);

            if (colliders.Length > 0)
            {
                Collider2D closestCollider = null;
                float closestDistance = float.MaxValue;

                foreach (Collider2D col in colliders)
                {
                    if (!col.gameObject.CompareTag("SnapPosition"))
                    {
                        // Calculate distance from mouse position to the collider's transform position
                        float distance = Vector2.Distance(mousePosition, col.transform.position);

                        // Find the closest collider
                        if (distance < closestDistance)
                        {
```

```
                            closestDistance = distance;
                            closestCollider = col;
                        }
                    }
                }
                if (closestCollider != null)
                {
                    selectedPointIndex = GetClosestPointIndex(mousePosition);
                    Debug.Log("Clicked on the closest GameObject: " + closestCollider.gameObject.name);
                }
            }
            if (selectedPointIndex != -1)
            {
                HighlightPoint(selectedPointIndex, 1);
            }
        }

        if (Input.GetMouseButton(0) && selectedPointIndex != -1)
        {
            Vector3 mousePosition = Camera.main.ScreenToWorldPoint(Input.mousePosition);
            mousePosition.z = 0;
            spline.SetPosition(selectedPointIndex, mousePosition);
            spriteShapeController.BakeMesh();
            UpdatePointIndicator(selectedPointIndex, mousePosition);
        }

        if (Input.GetMouseButtonUp(0))
        {
            if (selectedPointIndex != -1)
            {
                HighlightPoint(selectedPointIndex, 2);

            }
```

```
61                  selectedPointIndex = -1;
62              }
63          }
```

Listing 3.2: TouchHandleInput() method in C#, taken from the code of the video game

Given that the system utilizes single-touch inputs, touch interactions were treated as mouse inputs, meaning Unity's mouse methods were employed. As demonstrated in script 3.2, line 9, we utilized the function :
$Vector3 \quad mousePosition = Camera.main.ScreenToWorldPoint(Input.mousePosition)$
This function was used to modify the position of the nodes, aligning them with the position of the finger or the location where the click was registered. Additionally, the interaction zone for each node was expanded by assigning a collider to each one. In the event of a collision between node colliders, a function was implemented to calculate the distance between all the colliders in the interaction area, then select the node closest to the touch position.
To improve user visibility, a node changes its color to green upon interaction, as implemented in the script 3.2, lines 39–42.

3.4.2 Controller Input

For this study we have used the DualSense Wireless Controller (Figure 3.11), made by Sony PlayStation.

Although Unity provides a default input mapping through the input manager, it lacked clarity for our specific requirements. Therefore, we manually mapped the inputs that we needed to ensure accurate functionality.

The functionality assigned to each button is outlined below, based on the illustration in Figure 3.12:

Figure 3.11: The DualSense Wireless controller used in the study

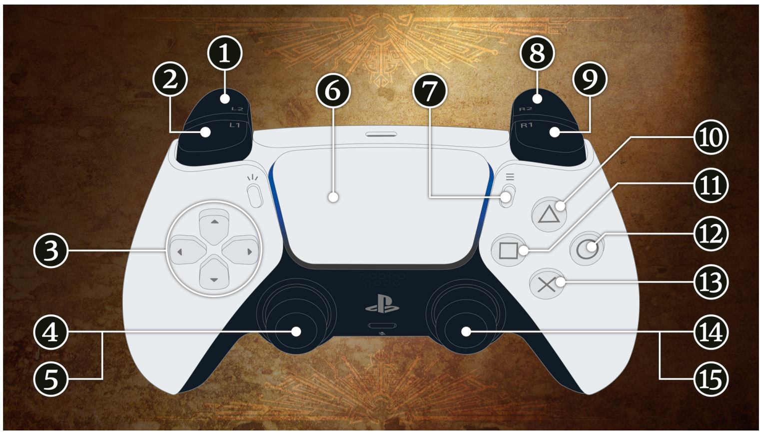


Figure 3.12: Image of a replica of the controller we used, with numerated buttons, image rights go to https://manual.capcom.com/mhwilds/beta/en/ps5/page/3/1

- **Button 1 (L2)**: Used to interact with buttons located on the top left corner of the screen, such as the "restart level" button, the "No" button(when the user is on the give-up canvas), and the "Controller" button when selecting an interaction method at the start of the user study.

- **Button 4 (left joystick)**: Used to interact with the shape by moving the selected node or to navigate between nodes.
- **Button 8 (R2)**: Used to interact with buttons on the top right corner of the screen (for example, to enlarge or reduce solution image, "Yes"(when the user is in the give up canvas), "Touch" when choosing an interaction method at the start of the user study, or "Next Level" after finishing a level or choosing to skip one).
- **Button 9 (R1)**: Used to interact with the "give up" button.
- **Button 13 (X)**: Used to select or deselect a node while interacting with the shape.

Due to the limited number of interaction options, the remaining controller buttons were not mapped or used.
Furthermore, we will discuss the two states that the controller offers when interacting with the shape:

- the navigation between nodes state
- the movement of nodes state

Which are influenced by similar inputs. As previously mentioned, when the user is in controller mode, a blue node appears (as shown in Figure 3.13), indicating the user's current position. In the "navigation between nodes" state, the player can navigate between the nodes (as demonstrated in Listing 3.3). By pushing the left joystick away from the user (upwards) the user selects the next node, which is the neighboring node to the current node in a clockwise direction. Conversely pulling the joystick towards the user (downwards), the user chooses the previous node (counter clockwise direction), as illustrated in Figure 3.13.

However, when the user presses the button X, the color of the node changes to green (as shown in listing 3.3, lines 1-7) which indicates that the user has entered the "movement of the node" state, (as outlined in Listing 3.4). In this state, the

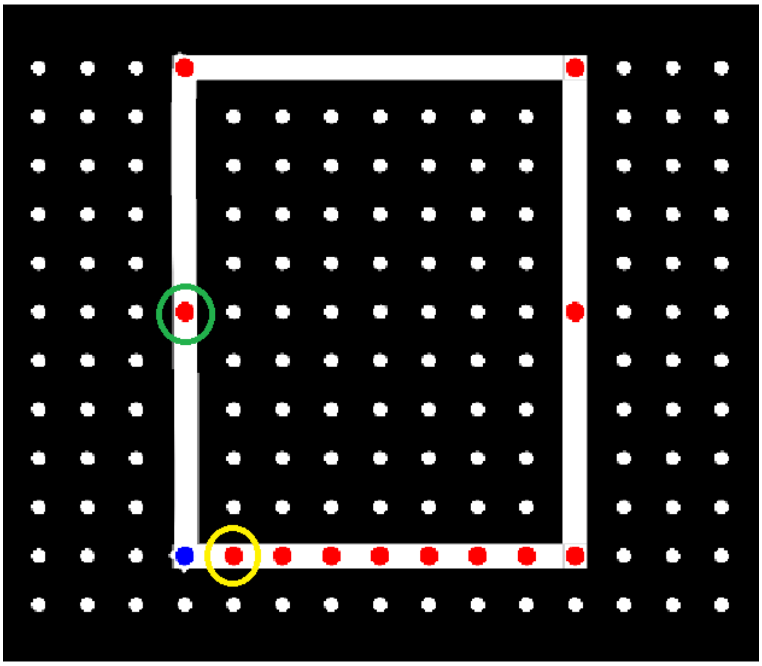


Figure 3.13: The node marked with the green circle implies the next node, and the node marked with the yellow circle implies the previous. The blue node is shown only in controller-input mode, which implies that the circle with the blue color is not selected but the player is currently at

left joystick controls the position of the green node, where the user can move the node by pushing the joystick towards the direction they want to move the node to. For instance pushing the joystick upwards the user will move the node continuously upwards, until the joystick is released. This behavior applies uniformly to all directions (left, up-left, up, up-right, right, down-right, down, down-left). As depicted in Listing 3.4 the position of the node changes based on which direction the user is pushing the joystick to, and is dependent on two components which are "MoveSpeed" and "Time.deltaTime", which are used to make the motion of the nodes smooth and continuous, rather than step-by-step move (grid-based movement), which requires constant pushing and releasing of the joystick.

```
 if (Input.GetButtonDown("1"))
        {
            IsSelected = !IsSelected;
            Debug.Log("IsSelected is:" + IsSelected);
            if (IsSelected) HighlightPoint(currentIndex, 1);
            else HighlightPoint(currentIndex, 3);
        }
if (!IsSelected)
        {
            if (Input.GetAxis("Vertical") < -0.5f && !IsNavigating)
            {
                // Get Previous Spline (Counter ClockWise)
                HighlightPoint(currentIndex, 2);
                currentIndex = ((currentIndex - 1 + pointPrefabs.Length) % pointPrefabs.Length);
                UpdateSelection();
            }
            else if (Input.GetAxis("Vertical") > 0.5f && !IsNavigating)
            {
                // Get Next Spline clockwise
                HighlightPoint(currentIndex, 2);
                currentIndex = (currentIndex + 1) % pointPrefabs.Length;
                UpdateSelection();
            }
            if (Input.GetAxis("Horizontal") > 0.5f && !IsNavigating) Debug.Log(" trying to move Right?");
            if (Input.GetAxis("Horizontal") < -0.5f && !IsNavigating) Debug.Log(" trying to move Left?");

            // Reset isNavigating when joystick is released
```

```
                if (Mathf.Abs(Input.GetAxis("Vertical")) < 0.5f)
                {
                    IsNavigating = false;
                }
            }
```

Listing 3.3: ControllerHandleInput() method in C#, taken from the code of the video game

```
if (IsSelected)
                {
                // If the joystick is outside the dead zone
                if (Mathf.Abs(Input.GetAxis("Vertical")) > 0.5f || Mathf.Abs(Input.GetAxis("Horizontal")) > 0.5f)
                {
                    Vector3 currentPosition = pointPrefabs[currentIndex].transform.position;
                    currentPosition.z = 0;

                    // Move vertically
                    currentPosition.y += Input.GetAxis("Vertical") * MoveSpeed * Time.deltaTime;

                    // Move horizontally
                    currentPosition.x += Input.GetAxis("Horizontal") * MoveSpeed * Time.deltaTime;

                    // Update the spline position and bake the mesh
                    spline.SetPosition(currentIndex, currentPosition);
                    spriteShapeController.BakeMesh();

                    // Update the point indicator
                    UpdatePointIndicator(currentIndex, currentPosition);

```

```
                    IsMoving = true;
                }
                else
                {
                    // Stop movement when the
                       joystick is in the dead
                       zone
                    IsMoving = false;
                }
            }
```

Listing 3.4: ControllerHandleInput() method in C#, taken from the code of the video game

3.5 User Study

3.5.1 Environment

The study was conduted in a controlled environment in a seminar room at the computer science department at the RWTH university of Aachen. A total of 11 Participants took part in the study, all individually with our supervision. The participants used a "iiyama Prolite XUB2793HS-B6" monitor that supports touch interaction, connected to a private laptop that had the video game prepared on it, and a DualSense Wireless Controller 3.11.

3.5.2 Process

For the user study, each participant was first presented with a consent form before starting the study. After signing the consent form, they were given a brief explanation of the study's process, followed by a demo of the video game to familiarize them with both interaction methods. Once they felt comfortable and ready, we started the study. At the start of the game, each participant was assigned a unique user ID and was recommended to begin with a specific interaction method, as previously explained, to minimize bias. During the study, relevant data was recorded and exported into a

.csv file at the end of each level. The collected data includes the following:

- **UserID**: An Identification number to attach the data to the user, with anonymity
- **LevelID**: The ID of each level to know which level the rest of the data belongs to
- **Input Method**: whether the level has been played with controller-inputs or touch-inputs
- **Try Number**: calculates how many tries the user needed to successfully or unsuccessfully finish a level.
- **Moves Counter**: depicts how many moves the user has made by the end of the level
- **Time**: shows us how much time the user needed to finish a level
- **Finished**: gives a binary output, which is whether the user has successfully finished a level or not.
- **Date and Time**: gives us a time stamp to the exact date and time the data were saved.

Eventually, each user was presented with a questionnaire about their experience with the game, and evaluate it.

Chapter 4

Evaluation

In this chapter we are going to talk about the results of the study and the limitations that we encountered.

4.1 Results

Throughout the game-play the time and the move count were calculated for each level across all the participants, those parameters will help us understand the difference between using each input method better. Furthermore the feedback from the quistionnaire gave us a clear result of the preference of the participants.

4.1.1 Game Results

In this section we will discuss the results of the video game, taking into consideration the number of moves users needed to finish levels and the amount of time they needed to finish the levels, and how often did users restart the levels.
As previously mentioned in Section 3.2, participants had the option to restart levels if needed. However out of 128 total attempts only 18 restarts were recorded. This suggests

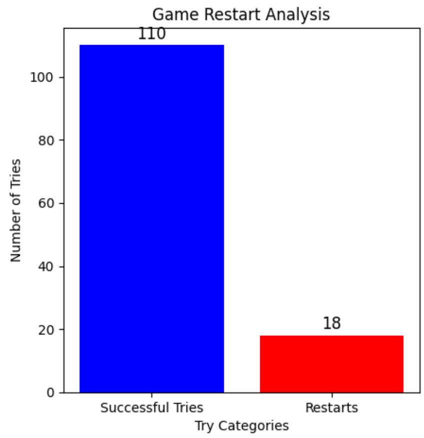


Figure 4.1: out of 128 tries altogether, only 18 restarts occurred which is as low as ~14% ,whereas 110 tries were finished successfully (meaning either completion of level, giving up, or failing due to time limit), which equals to ~86% of all tries

that participants preferred to continue with their progress ~86% of the times rather than restarting, to gain additional time (since each level try was limited by 5 minutes), as shown in Figure 4.1, out of the 18 restarts 15 were in controller-based and 3 in touch-based input.

Furthermore, when measuring the influence of the input method on each level of the 10 levels, we obtained the raw data that can be found in appendix B.
Since we have the same group of participants testing the game once with touch and another with controller inputs, we saw it fit to apply paired t-test (which has been introduced by William Sealy Gosset (Boland [2011]) in his published article titled "student"), on the collected data, to check whether the input method has an effect on the time and the moves count. To do so, we first had to filter the raw data to get aggregated data results, so we combined all tries that belong to the same "userID" and same "levelID"in the same row and then divided the data into 2 sections(controller and touch), after which we were able to calculate the p-values of the touch-input and the controller-input results.

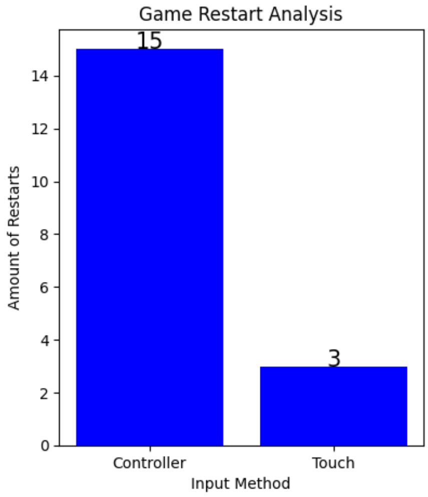


Figure 4.2: out of a total of 18 restarts, 15 were in controller-based and 3 in touch-based input

Paired t-test:

$$t = \frac{\bar{d}}{s_d/\sqrt{n}}$$

$$\bar{d} = \frac{\sum d_i}{n}$$

$$s_d = \sqrt{\frac{\sum(d_i - \bar{d})^2}{n-1}}$$

Where:

- **t** stands for: T-statistic, which tells us how significant the difference is.
- $\bar{d}$: the mean of the differences between the paired values.
- and s_d: Standard deviation of the differences.

Since our data was saved in a .csv file which is compatible with excel, we had the opportunity to utilize the built in function that offers the calculation of a paired t-test of given data. The command that was used is:

$= T.TEST(Xx : Xy, Yx : Yy, 2, 1)$

This function takes two columns, which we marked as X and Y (x and y stand for the first row of the data and the last row of the data), in the function the third component which is "2" indicates a two tailed test (since we are trying to find out whether there is a difference or not), and the fourth component ("1") indicates that the t-test is a paired t-test, the outcome of the paired t-test suggest the following:

- **p-value** $\leq$ **0.05**, There is a significant difference between the two conditions (controller-input and touch-input).
- **p-value** $\geq$ **0.05**, There is no significant difference between the two conditions.

When implemented with the time data we get the p-value of approximatly 0.00006, which is less than 0.05 suggesting that there is a significant difference between the two conditions.
When implemented with the moves count data we get the p-value of approximatly 0.4, which is greater than 0.05 suggesting that there is **no** significant difference between the two conditions.

4.1.2 User Evaluation

In this section we are going to discuss the evaluation the users gave the study through the questionnaire they were given after finishing the study.
Most of the participants were in their twenties, with the ages of the participants ranging between 23 and 30, with the majority being in their 25, while 8 of the participants were males and 3 females.

When asked whether the participants were familiar with video games, 45.5% of the participants said they were quite

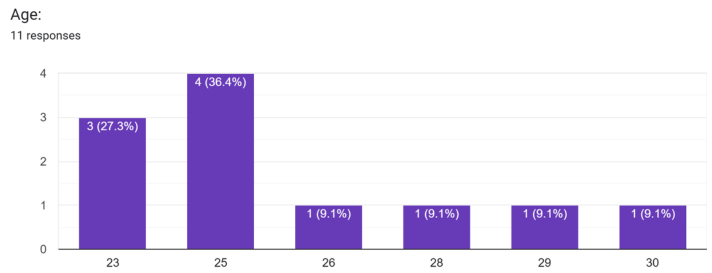


Figure 4.3: Ages of 11 participants that took part in the study

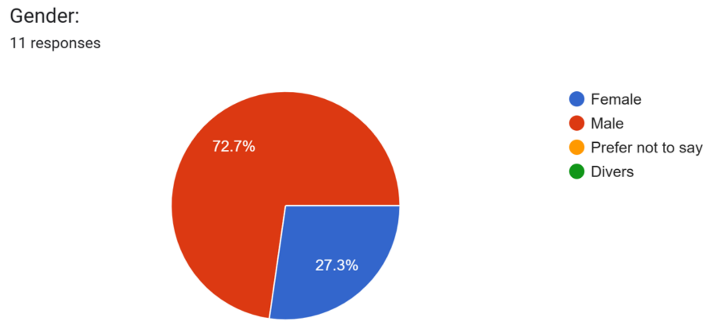


Figure 4.4: The genders the 11 participants identified themselves with

familiar with video games, while no one declared that they were unfamiliar, as shown in Figure 4.5.

When asked about the familiarity with touch-based interaction methods 72.7% of participants stated, that they are very familiar with it; 18.2% declared that they are familiar with it and only 9.1% said they were neutral, as shown in Figure 4.6.

Controller familiarity was very distributed, such that only 18.2% of participants were very familiar with controllers, 27.3% said they were slightly familiar, 18.2% were neu-

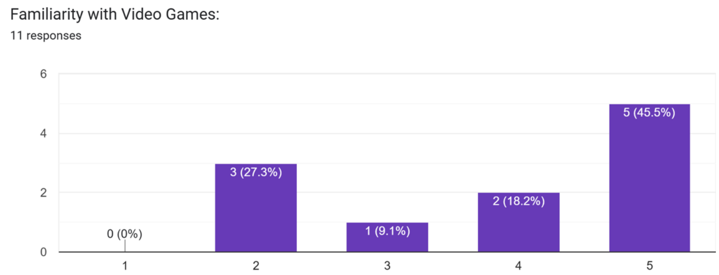


Figure 4.5: Familiarity with video games, from 1 (very unfamiliar) to 5 (very familiar)

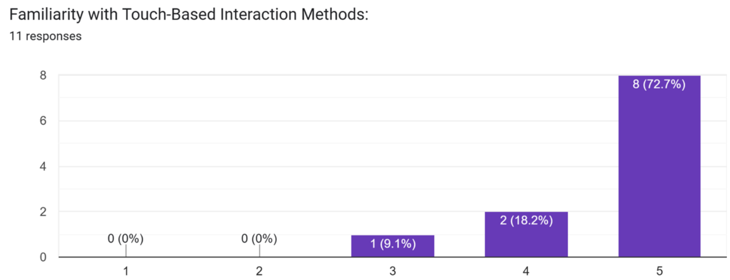


Figure 4.6: Familiarity with touch-based interaction methods, 1 being very unfamiliar and 5 being very familiar

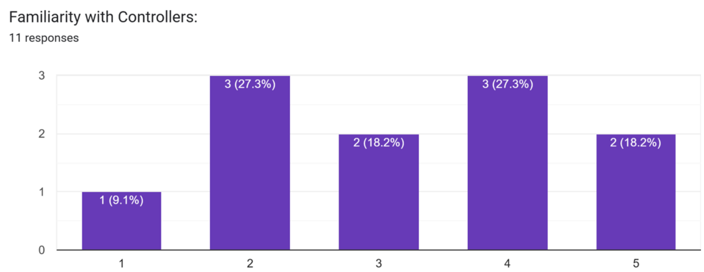


Figure 4.7: Familiarity with controllers chart from the questionnaire, 1 stands for unfamiliar, 2 slightly unfamiliar, 3 neutral, 4 familiar, and 5 very familiar

tral, 27.3% said they were slightly unfamiliar, and 9.1% stated they were unfamiliar with controllers.(see Figure 4.7)

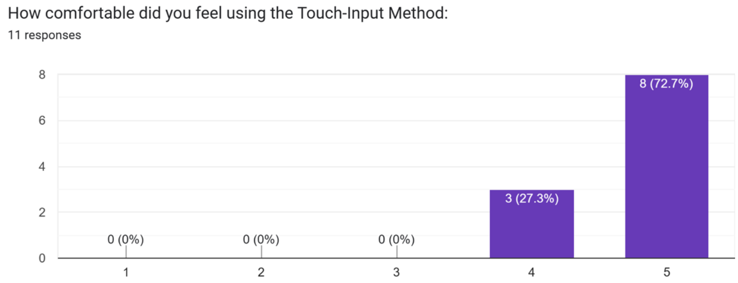


Figure 4.8: participants were asked to rate their comfort using touch on a scale from 1 to 5 (1 being uncomfortable and 5 being very comfortable)

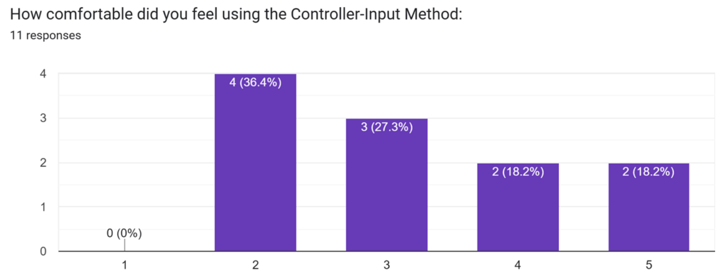


Figure 4.9: participants were asked to rate their comfort using controller on a scale from 1 to 5 (1 being uncomfortable and 5 being very comfortable)

Furthermore, when asked to rate their comfort using both methods with a scale from 1 to 5, 72% of the participants stated that they felt very comfortable(5) using touch-input, and 27.3% said they felt comfortable, whereas only 18.2% said that they felt very comfortable using controller-input and the majority (around 36.4%) stated that they felt slightly uncomfortable using the controller.

Moreover when the participants were asked which method they prefered, all of them stated that they favored using touch with 45.5% leaning a lot towards touch, and 55.5% slightly towards touch. Ultimately 81.8% declared that the instructions given at the start of the study were completely

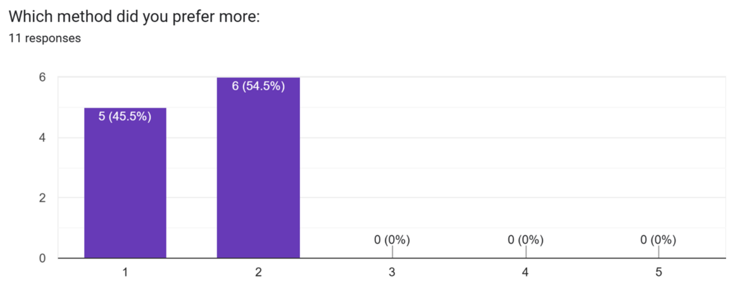


Figure 4.10: Participants were asked which input method they liked most on a scale from 1 being touch to 5 being controller

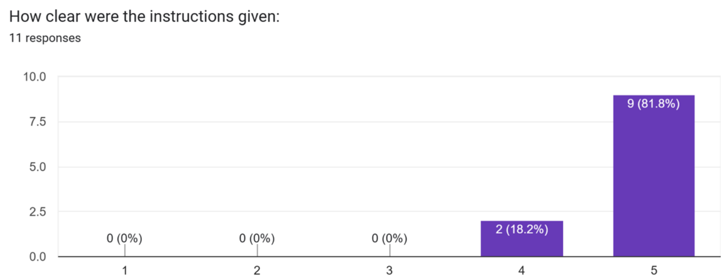


Figure 4.11: Participants were asked how clear were the instruction given at the start of the study on a scale from 1 being unclear to 5 being completely clear

clear, whereas 18.2% said that they were slightly clear.

Text Feedback

In addition to the above-mentioned mandatory questions, the users were given the option to write a textual feedback about the study altogether, as well as the chance to describe their likes and dislikes towards the video game. Most participants pointed out that they would have preferred being able to move the nodes from one position to the other (point-to-point movement or grid-based movement) while using controller, instead of having continuous movement, and snapping on release.

Additionally the feedback highlighted that touch-based input was generally preferred due to its intuitive nature and direct interaction, making node selection and placement easier. Many participants, especially those with little controller experience, found controller input challenging, as selecting and moving nodes required extra steps and was less precise (even after notifying the participants during the instruction phase at the start of the study about the selection process of the nodes in controller mode).
Several users also mentioned issues with image enlargement, including mismatches in scale and unclear visual guides. The overall game experience was enjoyable, with some finding controller input more engaging despite its challenges. Improvements such as clearer interaction feedback, and customizable controls.

4.2 Discussion

The results of this study indicate that touch-based input was significantly more efficient than controller-based iput when interacting with a large display, as participants required less time to complete levels when using touch-based input. This is enforced through the results of the paired t-test on the completion time (p = approximatly 0.00006), which says when a p-value is less than or equal to 0.05, it offers a significant difference between the two conditions that are being tested.
However, the number of moves required did not indicate any significant difference, with the p-value result of approximatly 0.4, meaning that input methods did not particullary affect the ability to perform under the tasks but rather only slowed execution of interactions due to navigation complexity. This may have been affected by the additional steps required when selecting and moving nodes with the controller, as users had to navigate between nodes before repositioning them, compared to the more intuitive and direct approach offered by the touch-based input.
Another key finding was the higher restart rate for controller input, with 83% of restarts occurring in controller mode. This suggests that users felt less confident in their

progress when using a controller, leading to more frequent attempts to optimize their approach. The lack of grid-based movement in the controller input method was frequently mentioned in user feedback, indicating that continuous movement may have contributed to this uncertainty. Implementing a snapping or grid-based movement system could potentially improve usability and reduce unnecessary restarts in future iterations of the game.

Furthermore, all participants ultimately favored touch input, reinforcing the idea that touch interaction is more natural for interactions with large displays. However, some participants with prior controller experience enjoyed the controller input despite its limitations. This suggests that controller input could still be a viable option if optimized for precision and ease of use. Additionally, prior experience with touch devices may have naturally favored touch input, as most participants were familiar with touch-based interactions, but less with controllers, as depicted in Figures 4.6 and 4.7.

Additionally, several users expressed their unease with node selection using the controller. The up and down (clockwise and counter clockwise) cycling through nodes was not always intuitive , which caused "a lot of misselection when choosing up and down depending on where the selection of the node was". Furthermore, participants noted that they "would have preferred having used D-pad" instead of the left joystick as a navigation tool, suggesting that the D-pad might have been a more effective alternative for selecting and moving nodes, as it provides a more structured directional input, enforcing the fact that a costumizable inputs could have boosted up the favorability towards the controller.

Moreover, some limitations of the game design added to the challenge. For example, some participants expressed the lack of an indication when switching interaction methods and level restarting, suggesting a possible improvement in that section. Additionally, participants shared their concerns regarding inconsistent scaling, when enlarging the images, one participant shared their thought, stating "Enlarging images would sometimes get confusing, as it doesn't always match the original image scale wise.". Addressing these issues could enhance the overall user experience by providing clearer visual cues and ensuring consistency in image scaling.

4.3 Limitations

In this section, we are going to talk about the limitations we encountered throughout the process of the thesis.
Some of the limitations were related to the development of the video game, for example, as previously mentioned in the text feedback of Section 4.1.2 some of the participants stated that the image scale, when the image is enlarged is different to that of the small image, that can be seen in Figure 3.5. This could have influenced the results of harder levels, since enlarging the image did not help the participants as much as intended. Another participant suggested a customizable controller -inputs to personalize the experience more and to increase comfort, also declaring that a point-to-point movement, might have increased the favorability towards controller, which was seen to be quite faint. Another limitation, has been observed during the user study with participant number 3, while completing "level_7", after the participant has successfully finished the level at exactly 5 minutes, the data has been registered twice in the .csv file, which could have manipulated the results if not discovered, this is where the completion indication has come in handy, due to the fact that when the situation occurred the winning canvas briefly emerged, then immediately after the game-over canvas showed, suggesting that an error has might have taken place at that exact moment, which we noted during the study and then checked the data, to confirm that it has indeed been double registered.
Furthermore the video-game, has not been pre-tested thoroughly, since a thorough bug and mistake testing would require a fair amount of testers and a relatively longer time, to check for mistakes, which means that the video-game could have had more mistakes than what has been discovered.
Some participants mentioned that the lack of indication between input method changing, has caused a bit of a surprise, which might have affected the performance of the

first level after switching the input method.
The controller used in the study does not support potential participants with upper limb disability as the paper Iacopetti et al. [2008] states. This shows that even if the controller exceeds touch in some aspects, it still lacks credibility among specific groups of possible users.

Chapter 5

Summary and Future Work

5.1 Summary and Contributions

To summarize, this thesis investigates the efficiency and usability of touch-based and controller-based input methods for interacting with large displays. To explore this comparison, a video game was developed that challenged participants to coordinate cognitive skills with hand movement and interaction with the display. 11 participants took part in the study. The participants had to complete 10 different levels, 5 of which with controller-based and another 5 with touch-based input. Performance was measured based on the time and number of node movement needed to complete the levels, with a time limit of 5 minutes.

The results of our study demonstrate that participants needed less time completing the levels when using touch-based input. This suggests that touch-based input is much more efficient when working with complex tasks. However, no notable difference was observed in the number of moves made, suggesting that the input method does not have an effect on the number of interactions a user makes. Furthermore, all participants favored the touch-input method, while some participants with controller experience did enjoy using the controller as well,

they did however state that changing the movement from continuous to grid-based could have improved the overall experience with the controller-input, and so would have a customizing option of the interaction keys.
Additionally, we observed minimal usage of the restart button, where 5 out of 11 participants did not use the button at all, and out of 128 combined tries only 18 restarts have been observed, and out of those 18 restarts a staggering ~83% occurred in controller-input mode.

5.2 Future Work

The video game developed primarily focuses on cognitive and complex tasks that require deeper thinking and problem-solving. However, it remains an open question whether the results would differ if a video game with less cognitive demands were used instead. For instance, a video game that tests the durability of the participants with the input methods, showing that some efficient methods might prove to be tiring and impractical when applied for longer periods of time.
While this study primarily focuses on drag-and-drop touch input, future research could explore a broader range of touch input methods simultaneously to gain a more comprehensive and general understanding of interaction efficiency and user experience with large displays. Additionally, expanding the study to include other touch-based input methods might increase preference towards hybrid interactions, in case some touch-based input methods prove to be less efficient than controller-based inputs, and vice versa.
Furthermore, we would like to see more uncommon methods that require less physical movement, for example, through eye-tracking as Drewes [2010] describes, or gesture-based input as Radkowski and Stritzke [2012] showed in their paper, which might decrease fatigue problems due to long interaction sessions.

Appendix A

Consent Form and Quistionnaire

Consent Form for Participation in a Research Study

Principal Investigator: Nasim Khatib

Institution: RWTH Aachen

E-Mail: nasim.khatib@rwth-aachen.de

Purpose of the Study

The purpose of this study is to compare touch and controller based input methods on a large display, through a video game.

Procedures

If you agree to participate:

1. You will take part in an approx. 45-minute study, that would require you to finish 5 levels with touch and 5 with controller
2. You will be asked to fill in a questionnaire, about your Interaction with the game

Confidentiality

- All data collected during this study will remain confidential.
- Your Data will be anonymized.

Voluntary Participation

Participation is voluntary. You may decline to answer any question or withdraw from the study at any time.

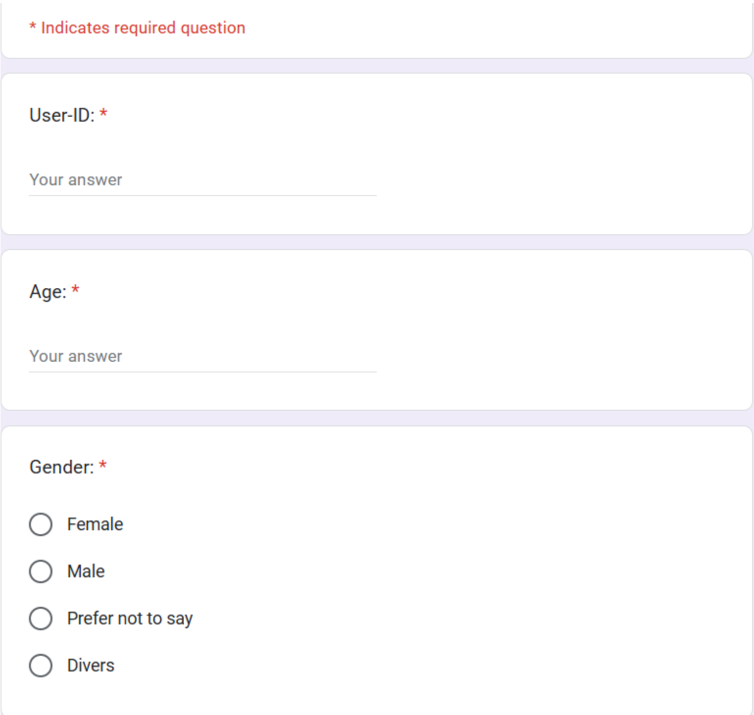
Participant Consent

Please read and sign below to indicate your agreement to participate.

- I have read and understood the information provided in this consent form.

Participant Name: ________________________________

Participant Signature: ______________________________ **Date:**_____________________________



* Indicates required question

User-ID: *

Your answer

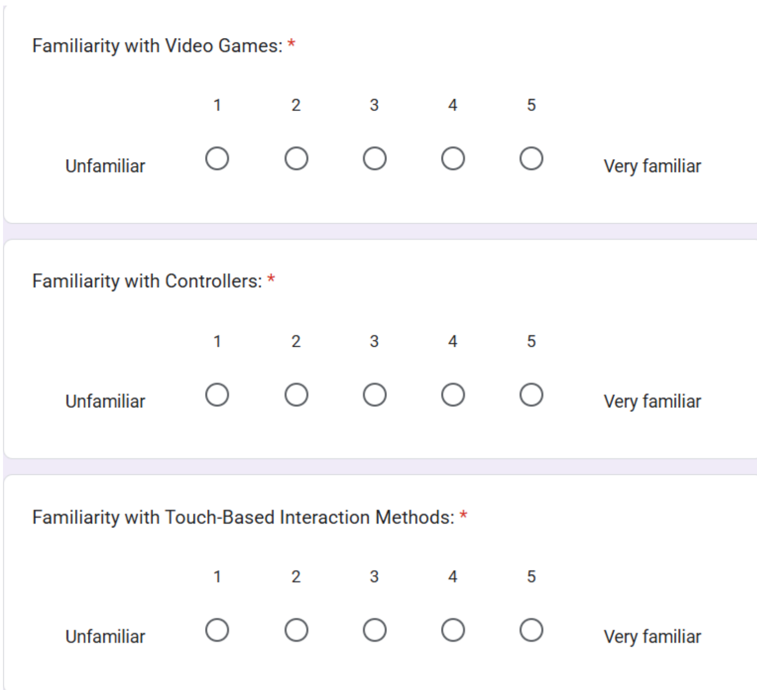
Age: *

Your answer

Gender: *

- Female
- Male
- Prefer not to say
- Divers

Figure A.1: Questionnaire page 1/4



Familiarity with Video Games: *

	1	2	3	4	5	
Unfamiliar	○	○	○	○	○	Very familiar

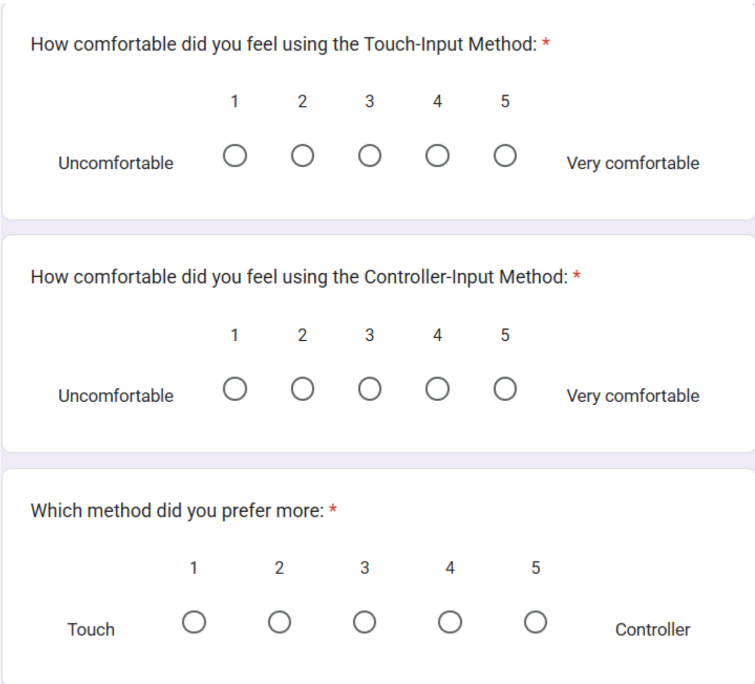
Familiarity with Controllers: *

	1	2	3	4	5	
Unfamiliar	○	○	○	○	○	Very familiar

Familiarity with Touch-Based Interaction Methods: *

	1	2	3	4	5	
Unfamiliar	○	○	○	○	○	Very familiar

Figure A.2: Questionnaire page 2/4



How comfortable did you feel using the Touch-Input Method: *

	1	2	3	4	5	
Uncomfortable	○	○	○	○	○	Very comfortable

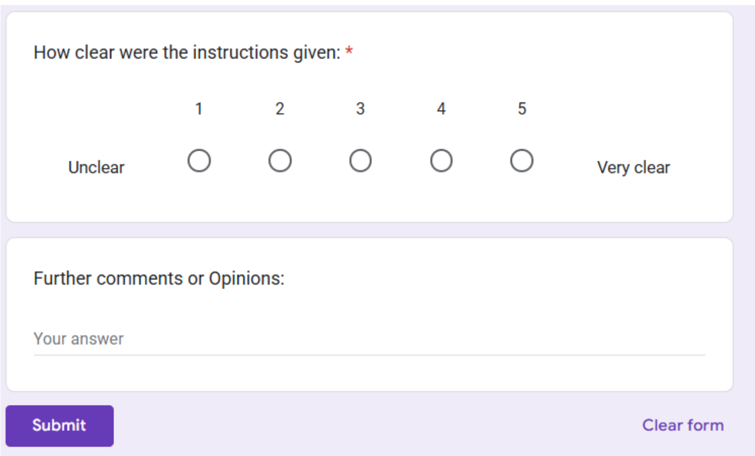
How comfortable did you feel using the Controller-Input Method: *

	1	2	3	4	5	
Uncomfortable	○	○	○	○	○	Very comfortable

Which method did you prefer more: *

	1	2	3	4	5	
Touch	○	○	○	○	○	Controller

Figure A.3: Questionnaire page 3/4



How clear were the instructions given: *

	1	2	3	4	5	
Unclear	○	○	○	○	○	Very clear

Further comments or Opinions:

Your answer

Submit

Clear form

Figure A.4: Questionnaire page 4/4

Appendix B

Raw Gameplay Data

1	UserID	LevelID	Input Method	Try Number	Moves Count	Time	Finished	Time Stamp
2	1	Level_3	Controller	1	0	00:28	RESTART	28.01.2025 10:14
3	1	Level_3	Controller	2	19	01:55	RESTART	28.01.2025 10:16
4	1	Level_3	Controller	3	2	00:26	RESTART	28.01.2025 10:16
5	1	Level_3	Controller	4	1	00:10	RESTART	28.01.2025 10:16
6	1	Level_3	Controller	5	18	02:13	True	28.01.2025 10:18
7	1	Level_4	Controller	1	13	01:39	True	28.01.2025 10:20
8	1	Level_6	Controller	1	1	00:37	RESTART	28.01.2025 10:21
9	1	Level_6	Controller	2	36	05:00	False	28.01.2025 10:26
10	1	Level_8	Controller	1	37	00:07	RESTART	28.01.2025 10:26
11	1	Level_8	Controller	2	42	03:29	True	28.01.2025 10:30
12	1	Level_10	Controller	1	1	00:24	RESTART	28.01.2025 10:30
13	1	Level_10	Controller	2	2	00:27	RESTART	28.01.2025 10:31
14	1	Level_10	Controller	3	38	05:00	False	28.01.2025 10:36
15	1	Level_1	Touch	1	5	00:20	RESTART	28.01.2025 10:39
16	1	Level_1	Touch	2	19	01:06	RESTART	28.01.2025 10:40
17	1	Level_1	Touch	3	5	00:24	True	28.01.2025 10:40
18	1	Level_2	Touch	1	8	00:39	True	28.01.2025 10:41
19	1	Level_7	Touch	1	44	02:59	True	28.01.2025 10:44
20	1	Level_5	Touch	1	48	03:09	True	28.01.2025 10:47
21	1	Level_9	Touch	1	64	05:00	False	28.01.2025 10:53
22	2	Level_1	Touch	1	12	00:50	True	28.01.2025 11:22
23	2	Level_2	Touch	1	5	00:13	True	28.01.2025 11:22
24	2	Level_7	Touch	1	29	01:35	True	28.01.2025 11:24
25	2	Level_6	Touch	1	33	01:27	True	28.01.2025 11:26

Figure B.1: Collected data 1/5

26	2	Level_10	Touch	1	25	01:15	True	28.01.2025 11:27
27	2	Level_3	Controller	1	12	01:43	True	28.01.2025 11:31
28	2	Level_4	Controller	1	12	01:12	True	28.01.2025 11:33
29	2	Level_5	Controller	1	28	03:59	True	28.01.2025 11:37
30	2	Level_8	Controller	1	22	03:18	True	28.01.2025 11:40
31	2	Level_9	Controller	1	27	05:00	False	28.01.2025 11:45
32	3	Level_1	Controller	1	16	01:15	True	28.01.2025 14:00
33	3	Level_3	Controller	1	18	01:17	True	28.01.2025 14:01
34	3	Level_7	Controller	1	59	05:00	False	28.01.2025 14:06
35	3	Level_7	Controller	1	59	05:00	False	28.01.2025 14:06
36	3	Level_5	Controller	1	21	01:49	True	28.01.2025 14:08
37	3	Level_9	Controller	1	58	05:00	False	28.01.2025 14:13
38	3	Level_4	Touch	1	8	00:18	True	28.01.2025 14:14
39	3	Level_2	Touch	1	7	00:14	True	28.01.2025 14:15
40	3	Level_6	Touch	1	71	02:44	True	28.01.2025 14:17
41	3	Level_8	Touch	1	41	01:28	True	28.01.2025 14:21
42	3	Level_10	Touch	1	87	05:00	False	28.01.2025 14:26
43	4	Level_4	Touch	1	8	00:22	True	30.01.2025 13:16
44	4	Level_2	Touch	1	5	00:10	True	30.01.2025 13:16
45	4	Level_7	Touch	1	14	00:35	True	30.01.2025 13:16
46	4	Level_5	Touch	1	60	02:17	True	30.01.2025 13:19
47	4	Level_9	Touch	1	50	02:14	True	30.01.2025 13:21
48	4	Level_1	Controller	1	18	01:20	True	30.01.2025 13:22
49	4	Level_3	Controller	1	16	01:15	True	30.01.2025 13:24
50	4	Level_8	Controller	1	18	01:24	True	30.01.2025 13:25

Figure B.2: Collected data 2/5, the red marked data in line 34, is the data collection error, which caused 1 set of data to be registered twice

51	4	Level_6	Controller	1	15	01:02	True	30.01.2025 13:26
52	4	Level_10	Controller	1	45	03:57	RESTART	30.01.2025 13:30
53	4	Level_10	Controller	2	75	05:00	False	30.01.2025 13:35
54	5	Level_1	Controller	1	6	00:33	True	30.01.2025 14:19
55	5	Level_4	Controller	1	38	04:43	True	30.01.2025 14:23
56	5	Level_8	Controller	1	12	01:23	True	30.01.2025 14:25
57	5	Level_6	Controller	1	19	03:02	RESTART	30.01.2025 14:28
58	5	Level_6	Controller	2	19	00:02	RESTART	30.01.2025 14:28
59	5	Level_6	Controller	3	38	05:00	False	30.01.2025 14:33
60	5	Level_10	Controller	1	23	03:10	True	30.01.2025 14:36
61	5	Level_3	Touch	1	12	00:48	True	30.01.2025 14:37
62	5	Level_2	Touch	1	27	00:50	True	30.01.2025 14:38
63	5	Level_7	Touch	1	75	04:20	True	30.01.2025 14:42
64	5	Level_5	Touch	1	5	00:12	True	30.01.2025 14:43
65	5	Level_9	Touch	1	27	01:24	RESTART	30.01.2025 14:44
66	5	Level_9	Touch	2	45	02:52	True	30.01.2025 14:47
67	6	Level_4	Touch	1	15	00:36	True	30.01.2025 15:19
68	6	Level_1	Touch	1	10	00:31	True	30.01.2025 15:19
69	6	Level_8	Touch	1	75	03:17	True	30.01.2025 15:23
70	6	Level_6	Touch	1	55	02:07	True	30.01.2025 15:25
71	6	Level_10	Touch	1	34	01:41	True	30.01.2025 15:26
72	6	Level_3	Controller	1	15	01:30	True	30.01.2025 15:29
73	6	Level_2	Controller	1	10	01:08	RESTART	30.01.2025 15:30
74	6	Level_2	Controller	2	23	01:51	True	30.01.2025 15:32
75	6	Level_5	Controller	1	30	03:27	True	30.01.2025 15:35

Figure B.3: Collected data 3/5

76	6	Level_7	Controller	1	31	03:34	True	30.01.2025 15:39
77	6	Level_9	Controller	1	37	04:17	True	30.01.2025 15:44
78	7	Level_4	Controller	1	9	00:59	True	04.02.2025 12:22
79	7	Level_3	Controller	1	11	01:24	True	04.02.2025 12:23
80	7	Level_8	Controller	1	22	02:32	True	04.02.2025 12:26
81	7	Level_6	Controller	1	32	04:20	True	04.02.2025 12:31
82	7	Level_9	Controller	1	40	05:00	False	04.02.2025 12:36
83	7	Level_2	Touch	1	6	00:16	True	04.02.2025 12:40
84	7	Level_1	Touch	1	13	00:28	True	04.02.2025 12:40
85	7	Level_5	Touch	1	17	00:45	True	04.02.2025 12:41
86	7	Level_7	Touch	1	27	01:32	True	04.02.2025 12:42
87	7	Level_10	Touch	1	104	05:00	False	04.02.2025 12:47
88	8	Level_2	Touch	1	7	00:21	True	04.02.2025 13:24
89	8	Level_3	Touch	1	11	00:27	True	04.02.2025 13:25
90	8	Level_5	Touch	1	12	01:05	True	04.02.2025 13:26
91	8	Level_8	Touch	1	19	01:00	True	04.02.2025 13:27
92	8	Level_10	Touch	1	55	04:51	True	04.02.2025 13:32
93	8	Level_1	Controller	1	8	01:40	True	04.02.2025 13:34
94	8	Level_4	Controller	1	6	00:47	True	04.02.2025 13:35
95	8	Level_6	Controller	1	24	05:00	False	04.02.2025 13:40
96	8	Level_7	Controller	1	24	05:00	False	04.02.2025 13:45
97	8	Level_9	Controller	1	24	05:00	False	04.02.2025 13:50
98	9	Level_2	Controller	1	8	00:41	True	04.02.2025 14:16
99	9	Level_4	Controller	1	6	00:42	True	04.02.2025 14:17
100	9	Level_5	Controller	1	50	05:00	False	04.02.2025 14:22

Figure B.4: Collected data 4/5

101	9	Level_8	Controller	1	25	03:01	True	04.02.2025 14:25
102	9	Level_10	Controller	1	8	01:08	RESTART	04.02.2025 14:26
103	9	Level_10	Controller	2	38	05:00	False	04.02.2025 14:31
104	9	Level_3	Touch	1	15	00:43	True	04.02.2025 14:32
105	9	Level_1	Touch	1	5	00:09	True	04.02.2025 14:32
106	9	Level_6	Touch	1	26	01:21	True	04.02.2025 14:34
107	9	Level_7	Touch	1	18	00:53	True	04.02.2025 14:35
108	9	Level_9	Touch	1	74	05:00	False	04.02.2025 14:40
109	10	Level_3	Touch	1	20	01:37	True	04.02.2025 15:25
110	10	Level_1	Touch	1	26	01:02	True	04.02.2025 15:27
111	10	Level_8	Touch	1	16	00:55	True	04.02.2025 15:28
112	10	Level_7	Touch	1	50	03:24	True	04.02.2025 15:31
113	10	Level_10	Touch	1	66	05:00	False	04.02.2025 15:37
114	10	Level_4	Controller	1	23	03:29	True	04.02.2025 15:41
115	10	Level_2	Controller	1	11	01:22	True	04.02.2025 15:42
116	10	Level_5	Controller	1	29	04:15	True	04.02.2025 15:47
117	10	Level_6	Controller	1	29	04:31	True	04.02.2025 15:51
118	10	Level_9	Controller	1	42	05:00	False	04.02.2025 15:56
119	11	Level_2	Controller	1	8	01:11	True	04.02.2025 17:22
120	11	Level_1	Controller	1	5	00:36	True	04.02.2025 17:23
121	11	Level_7	Controller	1	19	05:00	False	04.02.2025 17:28
122	11	Level_6	Controller	1	19	04:25	RESTART	04.02.2025 17:33
123	11	Level_6	Controller	2	15	02:25	True	04.02.2025 17:35
124	11	Level_10	Controller	1	26	04:32	RESTART	04.02.2025 17:40
125	11	Level_10	Controller	2	24	03:16	True	04.02.2025 17:43
126	11	Level_4	Touch	1	29	02:43	True	04.02.2025 17:46
127	11	Level_3	Touch	1	28	02:22	True	04.02.2025 17:48
128	11	Level_5	Touch	1	11	01:01	True	04.02.2025 17:49
129	11	Level_8	Touch	1	16	01:15	True	04.02.2025 17:50
130	11	Level_9	Touch	1	57	04:51	True	04.02.2025 17:55

Figure B.5: Collected data 5/5

Bibliography

[1] Carmelo Ardito, Paolo Buono, Maria Francesca Costabile, and Giuseppe Desolda. Interaction with large displays: A survey. *ACM Computing Surveys (CSUR)*, 47(3):1–38, 2015.

[2] Philip J Boland. William Sealy Gosset–An Inspiring 'Student'. In *Proc. 58th World Statistical Congress (Session STS028), Int. Statistical Inst., Dublin*, 2011.

[3] Costas Boletsis and Stian Kongsvik. Controller-based text-input techniques for virtual reality: An empirical comparison. 2019.

[4] Michelle A Brown and I Scott MacKenzie. Evaluating video game controller usability as related to user hand size. In *Proceedings of the International Conference on Multimedia and Human Computer Interaction*, pages 1–9, 2013.

[5] Alex Butler, Shahram Izadi, and Steve Hodges. SideSight: multi-" touch" interaction around small devices. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 201–204, 2008.

[6] Sebastian Doeweling and Urs Glaubitt. Drop-and-drag: easier drag & drop on large touchscreen displays. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*, pages 158–167, 2010.

[7] Heiko Drewes. *Eye gaze tracking for human computer interaction*. PhD thesis, lmu, 2010.

[8] Timothy Hoye and Joseph Kozak. Touch screens: A pressing technology. In *Tenth Annual Freshman Engineering Sustainability in the New Millennium Conference April*, volume 10, 2010.

[9] Fabrizio Iacopetti, Luca Fanucci, Roberto Roncella, David Giusti, and Andrea Scebba. Game console controller interface for people with disability. In *2008 International Conference on Complex, Intelligent and Software Intensive Systems*, pages 757–762. IEEE, 2008.

[10] Paul Kabbash, William Buxton, and Abigail Sellen. Two-handed input in a compound task. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 417–423, 1994.

[11] Minseok Kim and Jae Yeol Lee. Touch and hand gesture-based interactions for directly manipulating 3D virtual objects in mobile augmented reality. *Multimedia Tools and Applications*, 75:16529–16550, 2016.

[12] William Lu. Evolution of video game controllers. *Roseville: Prima Publishing*, 2, 2008.

[13] Bonnie MacKay, David Dearman, Kori Inkpen, and Carolyn Watters. Walk'n scroll: a comparison of software-based navigation techniques for different levels of mobility. In *Proceedings of the 7th international conference on Human computer interaction with mobile devices & services*, pages 183–190, 2005.

[14] Russell Owen, Gordon Kurtenbach, George Fitzmaurice, Thomas Baudel, and Bill Buxton. When it gets more difficult, use both hands: exploring bimanual curve manipulation. In *Proceedings of Graphics Interface 2005*, pages 17–24, 2005.

[15] Jeni Paay, Dimitrios Raptis, Jesper Kjeldskov, Mikael B Skov, Eric V Ruder, and Bjarke M Lauridsen. Investigating cross-device interaction between a handheld device and a large display. In *Proceedings of the 2017 chi conference on human factors in computing systems*, pages 6608–6619, 2017.

[16] Laxmi Pandey, Khalad Hasan, and Ahmed Sabbir Arif. Acceptability of speech and silent speech input methods in private and public. In *Proceedings of the 2021 CHI conference on human factors in computing systems*, pages 1–13, 2021.

[17] Rafael Radkowski and Christian Stritzke. Interactive hand gesture-based assembly for augmented reality applications. In *Proceedings of the 2012 International Conference on Advances in Computer-Human Interactions*, pages 303–308, 2012.

[18] Desney S Tan, Darren Gergle, Peter Scupelli, and Randy Pausch. Physically large displays improve performance on spatial tasks. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 13(1):71–99, 2006.

[19] Andrew D Wilson and Maneesh Agrawala. Text entry using a dual joystick game controller. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 475–478, 2006.

[20] Gareth W Young, Aidan Kehoe, and David Murphy. Usability testing of video game controllers. *Games User Research: A Case Study Approach*, pages 145–188, 2016.

[21] Loutfouz Zaman, Daniel Natapov, and Robert J Teather. Touchscreens vs. traditional controllers in handheld gaming. In *Proceedings of the international academic conference on the future of game design and technology*, pages 183–190, 2010.

Typeset March 17, 2025