A Qualitative Analysis and Framework for Well Designed Video Game Tutorials

MComp Computer Science

Bertie Taylor (170309406)

Supervisor: Dan Jackson

Word Count: 15,919 words

1st May 2021

Abstract

With the proliferation of video games within society, one key aspect to retaining a user base is their understanding of how to play a game. This dissertation explores the methods by which a designer can produce a tutorial for their game, first with an exploration of the surrounding literature including educational and psychological theories as well as more concrete suggestions for teaching methods. Then, implementation and testing of tutorialising methods is undertaken to ascertain the most useful and enjoyable methods. Finally, from this data, a framework is suggested to improve how tutorials are designed.

Declaration

“I declare that this dissertation represents my own work except where otherwise stated.”

Acknowledgments

Thanks go to my family for their support throughout my degree with special thanks to my sister for putting me on to the idea of dual coding theory.

Thanks also go to my supervisor for helping to guide this project – particularly helpful being the recommendation for think-aloud protocol

Finally, thanks go to my friends and housemates who have kept me going during this time without whom this dissertation may never have been finished.

Table of Contents

Table Of Figures

1 - Introduction

1.1 - Outlining the Problem

1.1.1 - The Context

Video Games are a form of multimedia that require active user interaction to be used and hence are a good area for research and understanding from an HCI standpoint. From an academic viewpoint, there has been some dismissal due to the mainly recreational focus that video games have taken however there has become more of a prevalence of use within education and health rehabilitation (Lohse et al., 2013). Even within a recreational context, it has been shown that among users' mental capabilities and physical skills such as hand-eye coordination have improved (Gozli et al., 2014). Knowing that there are benefits to the use of video games from a human perspective it is important to find the best methods to create new games and improve current ones. The video game tutorial is a key part of a good game experience. This is due to the fact that it is often the first part of a game a user will interact with, and therefore it must be engaging for all users as well as being efficient in teaching game mechanics to allow all users to reach proficiency for the harder tasks ahead.

1.1.2 - The Problem

Many users have mixed attitudes towards tutorials due to the large variance in quality found in current video games. In some games, a computer centred approach is taken- a user is given the bare minimum and their learning experience is unguided. This can lead to the user feeling frustrated and disengaged with the game. An example of this would be the "wall of text" simply an instruction sheet of the controls. Other games may take the opposite approach and become too heavy-handed. They will enforce a strict tutorial that means players who are already skilled become bored and once again disengaged with the game. The key to finding this middle ground is by finding a proven framework of ideas that produce game tutorials that keep the whole user base engaged and wanting to play. Existing work has suggested few ideas spread across large amounts of literature however with some collation this can be condensed. Currently, due to these issues, it is hard for someone working within the games industry to easily understand how they can produce a part of their game that is essential for a good video game experience.

1.1.3 - The Proposed Approach

In this project, an investigation will be carried out into current ideas surrounding tutorialisation. Literature on multimedia learning, game design, and user-centred design will be explored to collate these ideas. Then making use of the Unity Game Engine, a set of tutorials can be created to explore the methods by which these ideas can be implemented. This will enable a greater understanding of the ease by which these methods can be used- important in evaluating for producing the framework. Then using these tutorial's user testing will be conducted in which users will play the tutorials while participating in a think-aloud protocol to garner an understanding of what methods are aiding the player and which may be frustrating or boring them. With the data gathered from these last two stages, the best methods can be chosen to recommend a set of guidelines for video game designers to use that will improve the game experience for players.

1.2 - Project Aims & Objectives

The encompassing aim of this project is to produce a framework that explains how to create well-designed video game tutorials. This will be done by collecting data from three main sources. The first will be from existing literature that will suggest ideas. The second set of data will be an understanding of the complexity of implementation via creating tutorials for a 2D platformer. The final set of data will be based on user data gathered from user testing of these tutorials to

understand what tutorialisation methods put users at the best advantage to learn the game. My main aim has been broken down into the following objectives:

1) Identify at least 5 methods for tutorialisation in Games from exploring current literature

This objective will allow me to collect a selection of methods for use throughout the rest of my project. Here I will conduct background reading surrounding the project to understand the psychological and educational reasons why certain methods may be better than others in aiding tutorialisation.

2) Develop a series of at least 3 video game tutorials that each uses a combination of different tutorialising methods

This objective will aid two main parts of this project. Firstly, it will allow me to gather information on the ease by which each method can be implemented within a video game- important for evaluation. Secondly, these tutorials will be essential for use in the user research section of this project. The tutorials will be built upon the base of a 2D platformer that will also need to be implemented as part of this project. This genre of game has been chosen due to its skill-based nature which means it is easier to measure an increase in performance over something puzzle-based which may depend on prior intelligence. 2D platformers are also usually linear and not too complicated from a user perspective which will allow users to focus solely on how to play as opposed to being confused over where to go or what is going on (although solving these issues are key principles in game design also.) This will allow me to manage cognitive load more effectively in my tutorials.

3) Conduct user research on at least 5 participants using tutorials to discover the most efficient and enjoyable forms of tutorial

This objective will allow me to gain insight into how users interact with the chosen methods. Data from (Dworkin, 2012) has found that recommended numbers of participants in a qualitative study will range from 5 to 50 and hence I set 5 as my lower bound- especially given the social distancing restraints currently in place which will inhibit a larger scale study to be conducted with ease. The think-aloud protocol will be used to gather as much information on the thought processes of individuals as possible. This will enable me to understand the methods most useful to a user in a video game and hence be able to choose the best recommendations for a framework.

4) Analyse the research conducted and use the results to create the framework

This objective will allow me to complete the summative aim of this project by collating all of the research I have done and evaluating each piece's usefulness to help in deciding which methods are the most helpful in aiding and guiding the learning process in the beginning portion of a video game.

1.3 - Dissertation Structure

This Dissertation is broken down into 5 main sections. The following describes the information covered in each of these:

The first section is the introduction and within it, the domain of the problem is explained and the project aim is identified. This aim is broken down into a series of achievable objectives that come together to complete the project.

The second section is the background research. This section explores literature within the same area of research taking ideas from education, psychology, and the expected Human-computer interaction. Time is taken to study how tutorials have developed over time to have more context for the issues and the improvements that have been made. Ideas are then covered that detail the

potential methods suggested from prior research that could be used to tutorialise in games. This section aims to achieve the first objective.

The third section is the design and implementation of tutorials – covering the second objective of this project. The planning and specification are detailed through the use of a game design document and then the implementation of chosen tutorial methods is discussed and elaborated on – with suggestions for fast implementation methods developers can make use of in their games.

The fourth section explains the user research done making use of the tutorials developed in section three. It details the design of this research making use of the think-aloud protocol and then lays out the qualitative results making use of affinity mapping to analyse them and develop key insights into the best tutorialisation methods. This section achieves the third objective as laid out above.

Finally, the fifth section is an evaluation and conclusion of the project. First, the results of the research are compared against the background research to find congruency and differences. Then from this analysis, the final framework is suggested completing the fourth objective. To conclude, the achievement of the aim and objectives are discussed and improvements and further work are suggested for future research.

2. Background Research

2.1 What is A Video Game Tutorial

The first important idea to define is what exactly a tutorial in a video game is. In his book on tutorials (White, 2014) defines tutorials as follows:

“A tutorial is any component of a digital game that is intended to teach someone how to play”.

This gives a vague and mostly abstract definition from which to draw a multitude of ideas. It is helpful as a starting point and dictionary term; however, due to its concise format it does not give much context as to how this aim of teaching is achieved or why a tutorial should be included at all. The how is difficult to define and is hence the purpose of the dissertation as a whole, but the why is easier and is noted in the introduction. It can be seen that especially for new players including a tutorial in the game can increase the chances of continuous use and purchase intention (Moirn et al., 2016). This article also shows that for hardcore players there is a negligible difference in continuous use for hardcore/ experienced players when AB testing was conducted with and without tutorials. This gives the argument that from a designer’s point of view having a tutorial is important and useful in growing a userbase.

In (White, 2014) it is proposed that an important reason for including a tutorial is the increase of element interactivity in video games. This states that in modern games there are more elements a player must remember at any one time and hence this will make it more difficult for a player to form a model of how to play without some form of learning environment. The model that players form is known as a schema which is a set of structures within our mind used to organise information for fast recall. It is found that tutorials can be used to help build up schema to lessen the load on the mind later in the game meaning it is possible to handle the higher element interactivity in modern games.

White also defines two main types of tutorialisation, didactic and exploratory these help understand how a player learns. Didactic tutorialisation is instructional and tells a player to do certain tasks whereas in exploratory players choose their tasks and almost teach themselves the game. It is highlighted in (Kirschner et al., 2006) that exploratory methods alone “do not work” specifically in novice and intermediate learners but even for advanced learners it is often beneficial to have the guidance seen in a more didactic method. Hence it is a designer’s role to find the balance between didactic and exploratory methods when designing a video game tutorial.

2.2 The History of the Video Game Tutorial

Through the following research done into video game tutorials in commercial games, it can be defined that there are two key components to tutorialisation. First are the controls a player must use and the associated schemata for each. This tends to be a specific mechanic of the gameplay. Second, is the act of teaching the player how to use these controls within the context of the game. In modern games this tends to be integrated however in early games this was different. The first video games were arcade machines and they tended to have a simple dedicated control scheme for each game making them intuitive to use as the controller was often unique to the game. With the introduction of home systems, different games could be played using the same controller, and hence tutorials were necessary to understand the different game mechanics associated with each button.

Figure 1 – Super Mario Bros. 3 Instructions Manual ("Video Game Art & Tidbits", 2018)

Due to the importance of minimising the storage size of games (the original (Super Mario Bros., 1985) was around 40 kilobytes (LaCapria, 2019)), different methods were used to maximise teaching whilst keeping storage space down. As seen in the above figure for the third installment of the (Super Mario Bros., 1985) series a common method was to include a physical game manual along with the game cartridge. This contained the information there was no space for in the actual game including the first form of tutorial. Making use of images and text it showed controls and their related mechanics achieving the first component of the tutorial.

The second component is achieved within the actual game with the use of level design being key to helping a player understand the context as well as when different mechanics should be used. A popular example of this is again within (Super Mario Bros., 1985) with the first level seen to make use of techniques to guide the player in a way that does not feel constrained yet still teaches them how contextually the different mechanics work. The parts of this tutorial are discussed in many online videos and blogs with a good example being an article by (Iyer, 2017). Within they discuss the perceived affordances. As a player already knows the capabilities of the character, by placing a character off centre and facing in the direction of empty space it encourages the player to move into it where they view their first obstacles. This is clear as a design choice as Mario spends the rest of the game centred on the screen. Smart choices in design continue with the use of "signifiers" to represent explicit hints as to what an object could be what it could mean for the player.

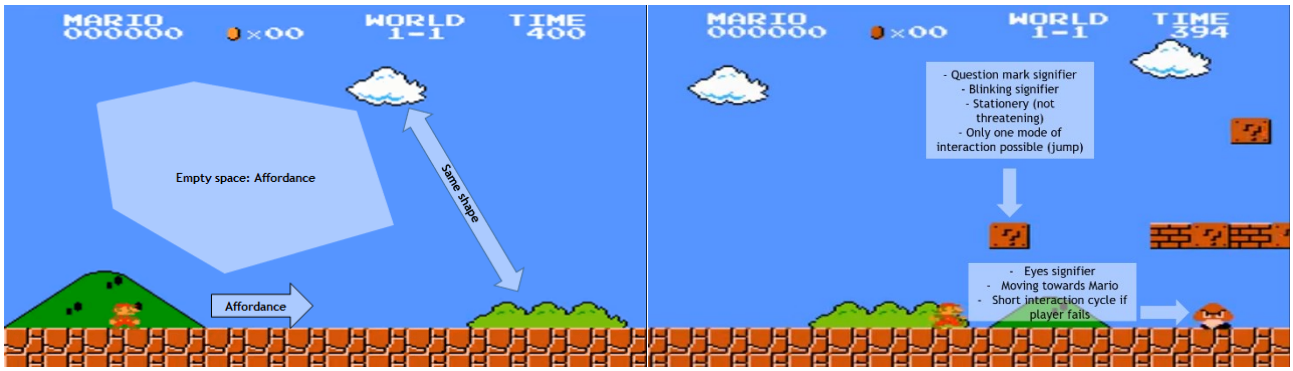


Figure 2: example of perceived affordances & signifier level design within Mario 1-1 (Iyer, 2017)

As spoken about in the introduction of his book (White, 2014) suggests that as video games have gained in the complexity of their design - due to the larger storage space and processing power available – So also has the amount of "element interactivity". Element interactivity is useful as a measure of how complex interacting a video game can be. It is suggested that in modern video games more focus must be maintained. For instance, in Pong (Alcorn, 1972) a player must watch

the ball and move the paddle to stop it from going off-screen. Whereas in the most recent Call of Duty title players must track the gun they're using how much ammo they have, where enemy players are, where they want to go and how to achieve this using the many complex controls. This is not a criticism of modern games as this higher element interactivity makes them more engaging and increases playtime and player interest. However, having more elements or mechanics means that better and more dedicated solutions to tutorialisation are needed today.

The 2000s saw one main type of tutorial, the optional tutorial level often still seen in real-time and turn-based strategy games. Here there are separate dedicated levels that make use of pop-ups through a guided scenario to teach the various complex mechanics of this type of game. These tutorials were rigid not allowing the player freedom hence many players decided to skip them and figure the games out themselves- often leaving them without essential knowledge. This lead to frustration among players who could not complete games due to lack of said knowledge and hence stopped playing. In action games of the time, a quicker and potentially even worse solution was used. The controls were shown to the player at the start often as a diagram with arrows relating mechanics to buttons on the controller. The first level would then simply turn down the difficulty from the rest of the game. This can leave the player feeling bored and unmotivated to continue playing.

As video games progressed more integrated tutorials have become commonplace. One idea is to make use of narrative to improve the tutorial as written about by (Bardzell et al., 2020). This journal paper found that the use of story helped encourage a player to keep on playing as wanting to find out what happens next is a good motivator. This is seen in more recent commercial games such as (Fallout 3, 2008) and (Horizon Zero Dawn, 2017) where the tutorial is disguised as a growing-up sequence as an explanation as to why they must learn to do things through the eyes of their character. This conceptual model of providing a story for learning is helpful but it can be seen throughout popular media that users still disdain the forced nature of this type of tutorial highlighted in the article by (Stout, 2015). One often-cited example of this done poorly is (Assassin's Creed III, 2012). In which the tutorial took players around 6 hours. This lead again to feelings of frustration or boredom in players who quit playing the game before they got to the main game due to this fault. This idea that tutorials are too long is one held by many experienced players for example seen in this popular blog post is this disdain concerning the aforementioned Assassin's Creed game (Metzger, 2010). The other issue players have with this type of tutorial is that it must be repeated when playing the game again ruining replayability and hence decreasing playtime this is evidenced many times including in this encyclopedia for media tropes ("Forced Tutorial - TV Tropes", 2021).

Video Game Designers have continued to learn from these ideas to produce their tutorials. The highest-selling video game of all time (Minecraft, 2011) ("List of best-selling video games - Wikipedia", 2021) did not have a tutorial in its classic version something occasionally praised by experienced players however this led to gatekeeping for new players. The more recent versions include useful tutorialisation in the form of contextualised tips as popups. These are designed to not be distracting appearing without obstructing gameplay. They also only occur at the moment a player needs them. For a modern platformer example, this is also seen in the game (Celeste, 2018) where these contextualised tips occur only when needed. The forced tutorial also only takes place within the first 5 minutes of the game meaning players do not feel constrained for a long time. The other good method (Celeste, 2018) makes use of is holding off on introducing some mechanics until much later in the game introducing 2-3 new mechanics in each part of the game and making use of level design to efficiently guide the player in learning them without linearly forcing them.

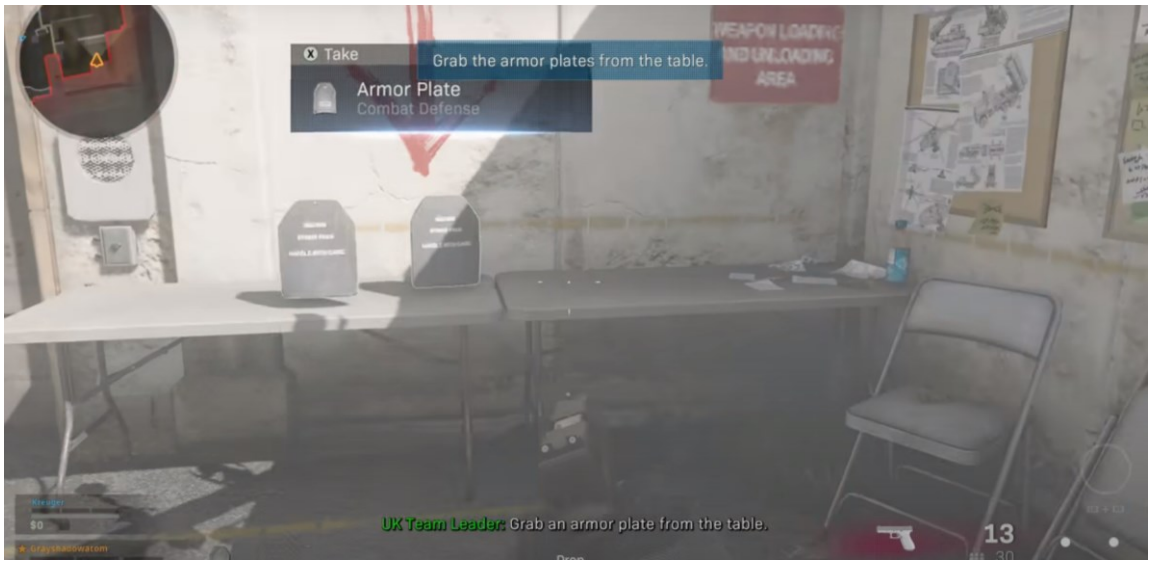


Figure 3: An example of the use of an audio tip contextualised for the moment when needed in Warzone ("Youtube: Call of Duty Warzone Tutorial", 2020)

In 2020, the popular game series Call of Duty released a free-to-play multiplayer game (Call of Duty: Warzone, 2020). This quickly gained over 50 million players (Statt, 2020), and with it being free to play this allowed new players to easily access the series for the first time. Therefore, a tutorial was key in helping new players understand how to play. It made use of a collection of methods, for example, audio was used to give tips something that is explored in 2.6 (visualised via subtitles in figure 3). The tutorial was also optional but makes sure that all first-time players are given the opportunity to play and even offers a reward as an incentive to complete it for experienced players who might need a reminder. Tips are also given in a contextualized manner meaning the player does not end up feeling confused by having all the information at once. Finally, the tutorial can be completed in under 5 minutes by an experienced player meaning that they will not begin to feel frustrated or bored as seen in other tutorials.

It can be seen how over time, tutorials have evolved along with video games to provide a better more effective experience for players but the methods they use to tutorialise need justification from a proven standpoint, and academic research in HCI, education, and psychology has been looked at to understand why different methods are used.

2.3 Cognitive Load and Overload Theory

Cognitive load theory is a theory defined by (Sweller, 1994) and built on since, most usefully in a conference paper defining helpful ideas in relation to online learning (Sweller, 2011). The theory puts forward that the human mind is considered in a similar way to a computer processor in that it has a short-term memory similar to RAM and a long-term memory similar to storage drives. The theory posits that the mind has a maximum load it can maintain at one time and if it receives more input it can lead to overload where no more information will be taken in. Three defined components can build up the load on the mind extraneous, endogenous, and germane. Extraneous load is the external items that can distract a learner such as people talking when a learner trying to focus. Endogenous load is the items relative to the task, the items you are focusing such as how does a player jump from one platform to the next. This will vary dependent on the complexity of the task at hand. The germane load is the subconscious process of short-term memory from the extraneous and endogenous load being transformed into long-term memory or “crystallized intelligence” (Cattell, 1971). It is also seen that as experience grows with a specific task- in the case of this paper a video game – then cognitive load is lessened and the learner can handle more complex tasks – this works similarly to activity theory (Kaptelinin, 2021), where activities become actions and

actions become operations as users spend more time using a technology. With video game tutorials, the purpose is to ensure that learning takes place in a way that does not send a learner/player into cognitive overload hence a player should not be given all the information at the same time and parts of the game should be introduced only when players have the competency to handle them without going into cognitive overload. The issue with a player going into cognitive overload will be either frustration as they feel overwhelmed or boredom as the overload distracts their mind and in both cases, the game will be turned off- the worst endpoint for a player from the designer's perspective.

2.4 Flow States

Flow is a state of mind first defined by (Csikszentmihalyi, 1990), it is defined as a feeling of intense focus on the task at hand often informally called being "in the zone". As (White, 2014) claims the causes of flow are complex and hard to predict however certain characteristics of a flow state can be defined as follows:

> *"Perfect balance between challenge and skill: Player is accomplishing tasks but using all of his or her energy to do so.*
>
> *Loss of self: Player becomes unaware of surroundings.*
>
> *Autotelism: Only observable in interview post-activity, player seems to want to continue activity for no reason other than the intrinsic value of the activity itself.*
>
> *Temporal distortion: Player becomes unaware of passage of time.*
>
> *Clear feedback: Player is able to relay the success/failure of his or her performance with perfect clarity.*
>
> *Loss of awareness: Player reports a feeling of acting by magic or automation to meet tasks and challenges.*
>
> *High self-efficacy: Player feels capable of meeting tasks assigned to him or her.*
>
> *Clear goals and outcomes: Player states he or she knew exactly how to achieve goals/outcomes."* (White, 2014)

Figure 4: The identifiers of a flow state (White, 2014)

There are three key points from this useful for video game tutorials. Firstly, clear feedback is important and easily achievable. The focus being that any tutorial should be responsive and adaptable and able to give help when it identifies a player is struggling- this will help to enter or maintain a flow state. Next, is clear goals and outcomes which will work in a very similar way in that a tutorial must be able to guide a player in knowing what they should or should not do without being didactic at all times as this can break the flow state. Finally, and often considered the most important is the perfect balance between challenge and skill. This balance is key to maintaining a flow state and as shown in the below figure if the balance is incorrect undesirable feelings can be reached by a player. If the challenge becomes too high while skill is too low a state of anxiety can be reached which can lead to the aforementioned cognitive overload which will lead to most likely frustration and "rage-quitting". As seen in experienced players during tutorials, if the skill level is high and the challenge level is low this can lead to boredom which will again lead to a player stopping playing.

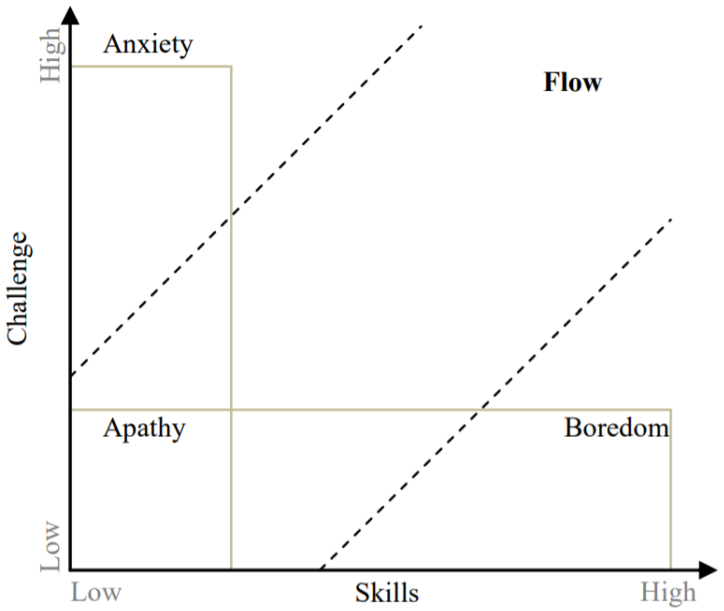


Figure 5: the flow balance diagram (Nacke & Lindley, 2008)

Flow happens in the balance with one caveat being that this does not mean that the levels of challenge and skill should be at their maximums but that they match each other. These variances can be explored through the ideas discussed by (Cai & Lin, 2011). Here it is defined that emotions can be defined by a combination of valence and arousal with frustration and boredom being the extremes of high and low arousal. (White, 2014) suggests that intense games such as first-person shooters tend to employ this high arousal state during game time but times of lower arousal included as breaks help keep a player from exhausting themselves or going into cognitive overload. Therefore, concerning tutorials it is important to balance the challenge and skill levels as well as the emotional arousal level to maintain a flow state and encourage the player to continue. Teaching mechanics in a way that doesn't interrupt this flow state is key and is important to increasing skill level with the challenge level and again keep the player focused and enjoying themselves.

2.5 Just in Time Instruction and Instructional Scaffolding

In (Gee, 2005) there is proposed a series of principles for learning used in some video games looked at from the point of view of introducing them to other learning environments. However, there is one idea that is popular in modern games still – the idea of Just in Time Instruction (JITI). The idea is to give players smaller parts of information that are contextually relevant at the moment they are needed. This moment of need tends to be defined by when a learner is struggling and in the context of game tutorials can be detected via the use of triggers and timers. This adaptability will provide a better tutorial as, (White, 2014) states an experienced player will not have these tutorial elements appear at all as they will not trigger them. This will improve the gameplay experience whilst ensuring that new players still get the help they need.

An alternative to JITI is instructional scaffolding an idea explored by (Tsai et al., 2013). This encourages an approach in which instruction is offered to all at first but if the learner is seen to succeed, instruction can be removed to allow an experience that does not distract an experienced player with tips. The other key part of scaffolding is that tips can reoccur if it is detected that a player is continuing to fail at a task. This continued support is useful in reminding players who come back to a game after a long time or new players who need reminding of certain mechanics.

(White, 2014) recommends the use of a combination of both these methods to produce a structured framework in which to deliver tutorials. I believe this to be sensible based on the educational theories already shown as giving tutorials as individual components is sure to lessen

cognitive load and giving tips in the moment will most likely help maintain a flow state due to the clear feedback criteria.

2.6 Dual Coding & Attentional Control Theory

One main principle in educational psychology is the idea that audio and visual information are processed separately and the cognitive cost of doing so is additive. This idea was originally suggested by (Clark & Paivio, 1991) with the proposition of dual coding theory. This theory posits that there are two key types of information taken in by the mind – "imagens" and "logogens". Imagens are visual stimuli such as images and video, whereas logogens are words – often thought of as audio as opposed to the written word which is mostly considered imagens. This idea has been developed upon countless times with a useful extension being the Attentional Control Theory of Multimedia Learning (Mann et al., 2002).

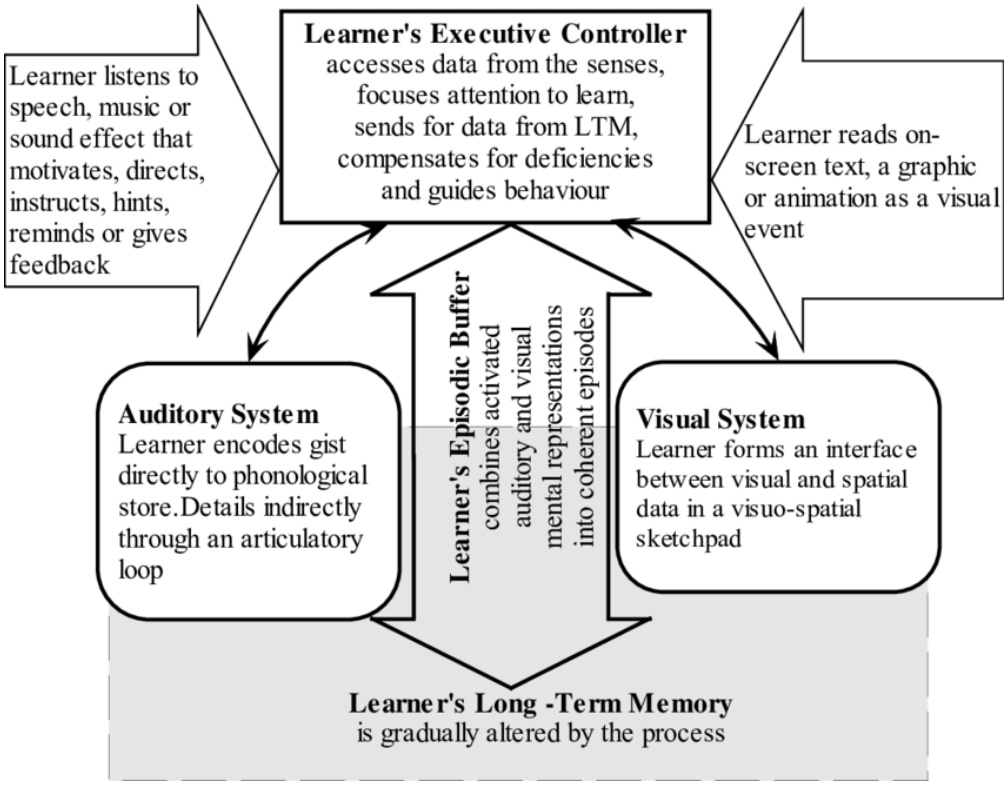


Figure 6: The Attentional Control Theory of Multimedia Learning Diagram (Mann, 2008)

As explained by Mann, Attentional Control Theory posits that learners take in information via the two senses sight and sound. It is different from dual coding in that it explicitly states sound as opposed to logogens. This I believe makes it more useful as a conceptual model for video game design, especially in tutorialisation. Mann also elaborates on the fact that the use of audio memory (also named gist memory) has been shown to improve recall of information as opposed to written instructions as seen in (Mann et al., 2012). From these two theories, it can be seen that audio has the potential to be an important tool for tutorialisation and should be tested further- with most research so far looking at more traditional forms of education (within schools).

2.7 Mayer's Principles of Multimedia Learning

Another theory that compounds on the idea of making use of audio and video to teach is Mayer's Cognitive Theory of Multimedia Learning (Mayer, 2009) – which explores the best methods for teaching in different media such as movies and video games through a set of principles. These principles are founded in psychological and neurological research into how the mind handles

learning new information when gathering it from a multimedia format. (White, 2014) analyses these principles and provides explanations for them in the context of video game tutorialisation.

The first principle to highlight is the multimedia principle which states that making use of both audio and video is important, this can be as simple as having the tip appear whilst a learner is playing to be able to also view exactly what is happening at any moment. Building upon this is the modality principle which encourages that speech is used for new information instead of text. This idea is one more recently adopted by commercial video games but has been proven to be more effective in keeping a learner in engaged – (White, 2012) states the effects of this were seen in an experiment done using World of Warcraft where he made use of this principle and found users with speech-based tutorials made fewer errors and showed higher game mastery than those with visual-only tutorials.

Next is the coherence principle which specifies that the player should not be overwhelmed by learning. (White, 2014) suggests this should be done by ensuring that the only things on screen should closely relate to the current learning objective (usually the mechanic being taught.) This closely relates to the idea of skill gating mentioned in online articles including (Laraman, 2018). The idea is that via the use of level design unavoidable obstacles can be placed in a player's way that can only be passed by using the mechanic they have just been taught. By making use of what they have been told in contextualised practice, a better understanding is gained of the relevant mechanic. This idea of skill gating enables the designer to ensure the player knows how to use a new skill whilst also helping the player to build confidence for when they need to use it later.

The segmenting principle is the idea that any complex idea should be broken down into smaller steps. In the context of video games, this is simple – each mechanic should have separate and dedicated training within the tutorial – as opposed to throwing a player into a difficult game level and giving them all the instructions at once. Giving time to develop and practice skills allows the learner to build understanding and confidence before trying harder levels- which is also consistent with flow states- increasing challenge as skill level increases. Mayer's pre-training principle follows similarly that a complex concept must be taught via a series of steps and be broken down to be better understood – (White, 2014) gives the example that instead of telling a player they have levelled up and can now choose new skills instead you must give them steps on opening menus, picking skills, and learning the new skill.

The next principle is the personalisation principle, which states that learners respond better to information given in a conversational tone as opposed to didactic. This means in the context of tutorials instruction should remain in context which helps keep the player engaged, with (White, 2014) stating that an instructional tone will incur additional cognitive load. Cognitive load can also be increased in other ways as stated in the redundancy principle. Here the idea is that when giving tips either audio or text should be used not both as the cost of using them is additive and one or the other is enough for understanding.

Mayer's principles offer many helpful and practical ideas for designing video game tutorials with more concrete rules than many of the other theories. The ideas offered here can be tested easily and provide a good structure for understanding what must be included with the earlier more abstract theories useful in backing these principles up.

2.8 Cognitive Apprenticeship

Another interesting idea presented for learning in educational psychology is put forward by (Collins et al., 1988), and proposes that learners will understand concepts and ideas better when they are guided by some form of expert, similar to the teacher-student relationship within a classroom. (Chang et al., 2013) explore this concept within the realms of game-based learning and identify a

framework for cognitive apprenticeship methods developed around a set of levels. These are demonstration, training, scaffolding, clarification, reflection, and exploration. Ideas such as training and scaffolding relate to ideas already discussed in this research section with instructional scaffolding and the segmenting principle, whereas the idea of demonstration is new. It is suggested that by a learner viewing an expert complete a task they will gain a better understanding of how to do so themselves. The ideas of reflection and exploration tend to be a natural subconscious part of learning already integrated into video games- as they are areas where active thought is necessary. (White, 2014) elaborates on the ideas of cognitive apprenticeship for tutorial design suggesting that the best way to include it is through the use of expert non-player character (NPC) characters often relying on state-based AI to give demonstration of key skills with the example of an NPC teammate who takes cover to show novice player the approach that should be taken in an FPS game.

2.9 A Note on Juiciness + Other General Game Design Frameworks

As some of form base game must be developed to create tutorials on top of, it is important to look at more general game design frameworks and rules. This is also useful in being able to determine what more general game design rules are and the rules that should be focused upon for tutorial design specifically.

One popular game design framework is MDA (Mechanics - Dynamics - Aesthetics) (Hunicke et al., 2001). It suggests that there are two key points a game can be viewed from either that of the player or designer. As a designer, you design mechanics (game rules) that define how the game works. Collections of these mechanics can lead to the evolution of dynamics for instance in an FPS, the mechanics of shooting and respawn points may lead to the dynamic of camping spawn points (Gallant, 2009). The aesthetics are the viewpoint from which a player sees the game and tend to focus on the feelings created from certain gameplay situations- in the case of the above example this feeling as the respawning player would most likely be frustration.

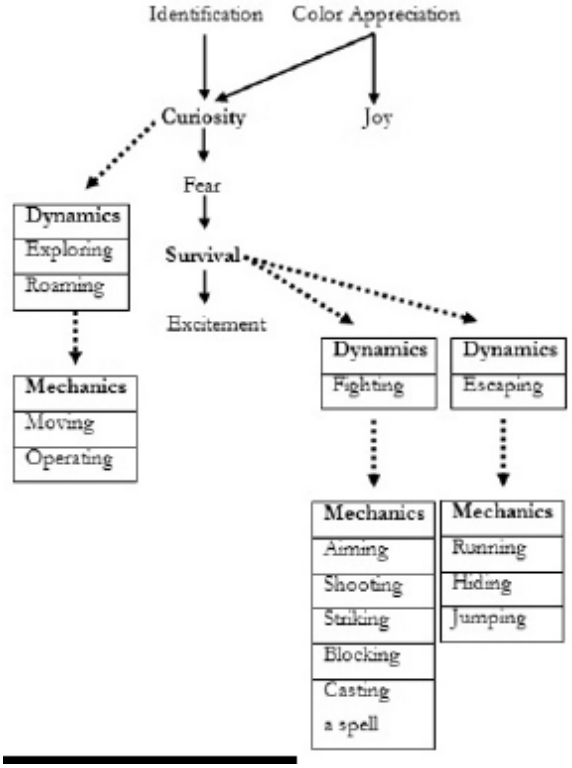


Figure 7: an example of the combined use of MDA and 6-11 frameworks to design interactions in a hypothetical adventure game (Dillon, 2010)

A criticism and improvement for MDA is made by (Dillon, 2010) with the 6-11 framework. It is proposed that MDA lacks clear definition of the state of mind of a player (despite its definition of the 8 types of fun). This extension suggests a series of 6 emotions and 11 instincts based on psychological research to identify the player's state of mind and be able to design games to guide a player through these specific emotions to encourage play. Through these two frameworks, it is possible to understand how designing interaction is possible as seen in the above figure 6. It is also easier to understand how a similar framework for tutorial design could be created from these ideas.

One other key idea to note is the idea of game feel – also known as juiciness as referred to by (Hicks et al., 2018). The term game feel - the title of a book by (Swink, 2008) is the idea of a set of theories that suggest micro-interactions that improve the way a player feels as they play. Ideas suggested in talks at the Dutch game garden including by (Nijman, 2013), are a selection of methods to improve game feel, including screenshake, particles, animation, and permanence. The development of game feel into juiciness happens when there are “large amounts of audiovisual feedback” used as the solutions to game feel. Through research with game designers and the use of affinity diagrams (Hicks et al., 2018) developed a framework for juicy design that suggests three high-level components of Game Characteristics, Game State, and Direct feedback. Each area has a few more concrete questions to improve juiciness and game feel within a game. The limitations however are that players were not involved in designing the framework- however, the ideas are still useful for developing the base game.

2.10 Summary

Through this section on background research, many things have been learned that will enable a framework to be produced. Important is the definition of a tutorial as any component of a game intended to teach a player to play. First explored were learning theories from education and psychology that offered up sound scientific reasoning for why certain design decisions would be made. These theories became more concrete throughout with Mayer’s principles being a highlight of ideas perfectly adaptable for use in games. The use of Gee’s JIT instruction and detail on the use of audio are also seen as highly useful in teaching a learner how to play. It can be seen through the history of game tutorials, how over time games have begun to inhabit the learning theories detailed here with the recent Call Of Duty title being an excellent example of methods such as JIT and audio tutorialisation. Finally, more general game design frameworks were explored as a note to understand the ideas that must be used when producing the base game to build tutorials on top of. This was also highly useful in understanding the format a game design framework must take- with this being the eventual goal of this dissertation project.

3. Design & Implementation of Tutorials

3.1 Introduction

This section details the methodology used to achieve the second objective of this dissertation. It will provide details on the design and implementation of a platformer base game, and its development to produce a set of tutorials following methods selected from the background research section. The methodology for introducing different tutorial styles will be detailed to gain a full understanding of the difficulty of implementation- something that is key for video game designers aiming to produce their own tutorials.

3.2 Planning & Implementation Workflow

For the implementation of tutorials, an Agile development method was chosen. As described by (Workfront, 2021) an Agile development cycle makes use of short iterative steps named sprints in which to complete work. One of the four core values central to Agile is responding to change as stated in (Keith, 2010). This is useful in a project where it is unsure how long each developmental stage may take. This adaptable model is also useful in a design project such as this one that follows an exploratory approach as Agile allows design decisions to be changed later within the process.

Through exploring suggestions from (Starloop Studios, 2020) there was a recommendation for using a Kanban board as a way to organise sprints for a video game project. (Trello, 2021) was chosen as the software to be used to organise this project making use of recommended layout for a kanban board as defined in (Joiner, 2018). It was decided to use a weekly sprint, where within each week a specific area of the development would be addressed with the benefit of agile being that unfinished tasks at the end of a sprint could be reallocated into later weeks. The implementation was completed over six sprints beginning with the development of a base game and moving to the implementation of a variety of tutorial methods in the later weeks. This relatively short development cycle ensured plenty of time for user research and analysis an objective of the project considered more important to the final framework.

3.3 Tools and Technologies

Within, the implementation there is a multitude of tools used for different purposes. The below section explains the purpose of each tool and why it was chosen for use in this project.

3.3.1 Unity Game Engine & Editor

A game engine is a key piece of the toolkit when developing a video game. It is a piece of software that contains a rendering pipeline, physics engine, sound engine, memory management, and scripting among other features. One option for this project is to build this engine from the ground up however with the focus being on the higher-level concepts and user research it was decided that a prebuilt engine should be used to save time and ensure there would be less error in the game itself. (Unity, 2019) was chosen as the engine for the following reasons:

- Unity provides all the core functionalities expected of a game engine to a high level including an easy to use UI development system important for use in tutorials
- The Unity engine is free for use in commercial and educational contexts
- Due to this, there is a large widespread community- which means there are a multitude of resources for troubleshooting and using the engine.
- There is a provided editor with a well-designed GUI which promotes fast development- specifically Unity makes use of a Tilemap system for 2D level design which can allow rapid development at this stage

- Unity allows games to be exported easily to multiple platforms including most operating systems and the web.
- Unity offers custom scripting support for multiple languages – which will make it easy to add the functionality needed into the game- without having to follow the more rigid workflow of a visual scripting design as seen in other engines.

3.3.2 C# & Unity Script

The two most popular languages used for scripting within Unity are Unity Script and C#. Unity Script is a custom version of Javascript that has the add-on of classes however Unity technologies have now stated they will deprecate this language and hence is not the best option. C# is the best-supported language for development with all Unity resources giving example code in this language. It offers the functionality of other languages in the C family- such as pointers and being compiled as opposed to interpreted, which are highly useful in game development whilst also making use of dynamic garbage collection and better error handling. C# is object-oriented which is key and offers good support for the .NET framework used by Microsoft for Unity.

3.3.3 Visual Studio

An IDE (Integrated Development Environment) needed to be chosen for coding custom scripts, in this case, the recommended environment for Unity is Microsoft Visual Studio. This is a good choice as it is fully featured and offers good debugging tools as well as excellent plugins for integration with Unity. These tools enable faster development with features such as IntelliSense code integration for code completion, advanced syntax highlighting, and “Superior Debugging” (Miller, 2019).

3.3.4 Game Assets

Assets for the game were sourced from a variety of sources with all pre-created assets being used following the Creative Commons license (Creative Commons, 2021). Game tiles were sourced from the Unity Asset Store, and game sounds and music were sourced from ("ZapSplat", 2021). The free open-source software GIMP (Kimball & Mattis, 2021) was used to produce custom visual assets such as the player character, and the sound recording and editing software Audacity (Audacity Team, 2021) was used for voice lines and other custom game sounds.

3.3.5 GitHub & Pages

GitHub was used as a repository to store versions of the project with the Pages feature being used as a way to host the final project online for free so participants could easily access it for testing. Pages do not offer dynamic server-side scripting such as PHP, however, this is not needed for this project so this free service is perfect for the required purpose.

3.4 Game Design Document

Now that the tools and technologies have been defined, the base game itself is presented to explain what players will be playing. In game development, a rigid requirement specification is not used often with the use of a Game Design Document being commonplace. The following section defines the idea of the game as well as how it is played.

The game is a simple 2D platformer – a genre of game defined by the jumping and movement mechanics with the aim of moving from one point to another. The player can succeed in the game by moving through each level starting on the left and exiting on the right with the aim of clearing all the levels. Players fail when they hit spikes throughout the game world causing them to die and respawn at the start of that level.

The game will use a simple pixel-based art style- with sprites of a 16x16 nature. This will mean the game loads quickly due to assets not being of large file size- it also promotes a nostalgic feel reminiscent of other platformers which will aim to help the player remember associated schema from other platformers through familiarity. Background music will provide a calming and relaxing experience without being distracting and will hence be instrumental and minimal without sharp sounds. In reference to theme, the character will be traveling through a jungle reminding players of schema relating to exploring which will encourage players to want to explore new levels and keep them moving forward. The player character is a simple white sprite with full movement animations- this allows them to easily stand out from the background and easily be focused on by the player.

3.5 Explanation & Implementation of Mechanics

The below table explains the game mechanics and the controls a player must use to manipulate them. The aim was to implement a complex movement system due to the fact sufficient material was needed for tutorialisation. This led to two types of mechanic- The first being the player mechanics and the second- environment mechanics. Player mechanics (1-6) were all usable via direct interaction with the controls on the keyboard- with some buttons doing different things within different contexts. Environment mechanics (7-10) were ideas that would affect the player as they traveled through the levels and relied on the assets that made up the world.

Game Mechanic	How Does it Work?
1) Horizontal Movement	Use of Left and Right arrow keys to move in the x-direction
2) Jumping	Use of the C button to create vertical movement when touching the ground
3) Double Jumping	Use of the C button whilst in mid-air to perform a boost allowing the player to jump further
4) Wall Grabbing	Use of the Z button when against a wall to allow a player to hold on and prevent from falling. The player lets go by releasing the Z button
5) Wall Climbing	Use of the Up and Down arrow keys while holding a wall to move vertically up and down the wall
6) Wall Jumping	Use of the C button whilst touching a wall does a jump that launches the player diagonally away from the wall being touched – the player must not be holding the Z button at the time they jump
7) Spikes	If a player falls on spikes this will result in player death and respawn
8) Moving Platforms	These are platforms that move in a predictable direction (back and forth) to allow a player to move further distances. (Important note: while a player is on a moving platform their position moves with it)
9) Crumbling Platforms	These are platforms that when landed on by a player will quickly crumble away and vanish leaving the player to fall unless they jump off again quickly
10) Bounce pads	These visually occur as mushrooms that when landed on provide a boosted jump to allow the player to achieve more height over large walls or gaps

Figure 8: A table detailing the game mechanics and their workings

The mechanics were all implemented upon a character controller script attached to the player game object. The input is read in via Unity's built-in system and following this, the player's movement is affected. A brief explanation of the implementation of mechanics follows:

First for player mechanics was the horizontal movement, done via constant velocity allowing a tight control feel to this- giving the player more control than if pressing arrow keys affected acceleration. The vertical movement however was done via the manipulation of forces which allowed more realistic jumping affected via a varying gravity coefficient which produced the style of jump commonly associated with platformers and detailed here ("A Perfect Jump in Unity - A Complete Guide", 2019). The jumping functionality was adapted for use with the double jump and wall jump methods with wall jumping making use of a directional force pushing the player away from the wall they were on. This wall detection was done via the use of raycasting and with checks for input, this also allowed the wall grabbing functionality which set the gravity coefficient to 0 in the case "Z" was pressed while on a wall. Wall climbing was then simply implemented by changing the player's y velocity via the Up and Down arrow key presses. A wall slide was implemented where if the player was on a wall but not pressing "Z" they would slowly slide down it- this would allow players in precarious positions to make changes to their movement with more time.

Concerning environmental mechanics, the first and key to the game were spikes- this was addressed by setting the player to a dead state when they collided with a spike collider- which would then reload the current game level. Large box colliders on each end of a level did a similar job for moving the player to the previous or next level accordingly. Moving platforms contained their own scripts which manipulated their movement according to a set velocity. When players collided with the top of a moving platform the platform is made their parent which influences the player's relative position until the player jumps off the platform and is removed from the child list of the relative moving platform. Mechanics were also combined here with the inclusion of spiked moving platforms which would enable testing on seeing how players adapted to combining schema. In implementation, this simply involved adding spikes to the moving platform model with the associated tagged colliders. Crumbling platforms contained a simple script that detected when a player landed upon them via a collider- when this happens a crumbling animation is run followed by turning off the collider of the platform for a limited time and then resetting it. Finally, bounce pads work in a similar way to spikes with detection in the player controller that detects when a player's feet collide with the top of a mushroom. In that case, the jump function is called automatically with a larger vertical force giving the large boost seen in this mechanic.

3.6 Level Design

Designing levels is a key part of the tutorial as evidenced by specific principles shown in the background research. (Mayer, 2009) provides good guidance with suggestions such as the pre-training and segmenting principles and these ideas of teaching each mechanic individually should be tested. Hence for this project, a series of levels were designed that introduced each mechanic this way with the final level integrating all the mechanics – this meant this final level (level 8) could also be used as an individual short tutorial to allow simple AB testing on whether a longer or shorter tutorial is better. The below table gives a description of each level and the mechanics it aims to teach- with appendix A providing images of the finished level designs.

Level No	What Does it look Like?	Which Mechanics Does it teach?
1	Simple level with raised section and one spike pit.	Horizontal Movement, Spikes Jumping
2	Similar Level with two spike pits one larger that cannot be made across with normal jump	Double jumping and reinforces normal jumping with the first pit
3	Large spike pit 3 bounce pads along it allow you to get across easily- one of which must use double jump to reach the end	Bounce pads
4	Level with two spike pits first with one crumbling platform then larger pit with multiple platforms	Crumbling platforms and reinforces jumping mechanics
5	Almost vertical mirror of the previous level - small pit to remind players of jumping then larger pit with a large wall for player to grab and climb up	Wall Grabbing and climbing
6	Floating block with spikes on left to stop climbing and overhangs on the right to promote wall jumping.	Wall jumping
7	Large spike pit with two moving platforms above- one with spikes	Moving Platforms + Spiked Moving Platforms
8	Combination of mechanics to test player and give tips when they are struggling – camera moves with the player on this longer level	All Mechanics

Figure 9: A table detailing the levels designed for this project

As can be seen above, it was decided that some mechanics should be integrated into the same level, for instance, the first level introduces three mechanics however they are taught individually throughout the level with a focus on implementing Mayer's coherence principle through the idea of skill gating. In the first level, this is done by the design choice of a small raised platform blocking the player's path to the exit- this enforces that the player must use the jumping mechanic to proceed ensuring that the player knows how to use it. Simple design choices like this are used throughout the levels with another key example being level 6. Here to skill gate – spikes are added to the left-hand side of the central platform to stop the player from using the already learned climbing mechanic. Overhangs on the right also prevent climbing from being used and hence ensure the player must wall jump to reach the exit.

Another important design idea considered was flow – with the idea that it must be clear where the player must go- this ties into the idea of perceived affordances mentioned by (Iyer, 2017). It was decided that simply leaving a hole in the right side of the level was sufficient to show an exit while ensuring the rest of the level was surrounded by walls or spikes this ensured players knew there was only one way out. Another key part of flow is the literal definition for the player of feeling as if the player movement flows well – one good example of this in the level design is in level 3. This is due to the placement of bounce pads at perfect intervals to allow the player to reach the far side of the level by simply keeping hold of the right arrow key – this good feeling helps to bring a player into or maintain a flow state.

One final idea that was important to design was to ensure that already taught mechanics were reinforced. (White, 2014) states that developing schema is an iterative process and hence players must be encouraged to reuse mechanics they already know to do two main things, first to reinforce learning and second to build confidence. As evidenced by exploratory tutorialisation methods, being able to reuse the schema already built allows a player to improve their knowledge further.

For implementation, the Unity tilemap system was used which allowed levels to be developed quickly by simply painting the chosen asset tiles into the scene. Decorative elements and spikes were added on different layers to add depth and allow easier differentiation when testing for collisions. The in-built tilemap colliders were used to represent the tilemap layers within the Unity physics system.

3.7 Explanation of Tutorial Decisions

At this point, the base game for the project was completed with the levels also fully designed following ideas from the background research. From here it is key to identify the different types of tutorial that will be tested and hence the decisions that were made on tutorials which should be developed.

Already, through the level design, one testing idea has been introduced with the idea of a short tutorial vs a long tutorial. The expected difference predicted here is that the short tutorial won't allow players sufficient time to develop schema and hence they will fail more times than a player who has built up confidence through learning in a longer tutorial.

One other key idea tested was the recommendation from (Gee, 2005) for the use of Just in Time instruction. To test this two key types of tutorial must be developed. The first gives all game instruction at the start of the game (similar to game manuals in early games). The second will offer just-in-time instruction as explained by (White, 2014) via the use of the collection of data on user behaviour to be able to offer hints to the player at times it is believed to be needed. It is predicted that – following Gee- just in time instruction will be viewed more favourably by participants due to it being less likely to induce cognitive overload with the spreading out of information also recommended in the segmenting principle (Mayer, 2009).

The next idea tested is the use of dual coding/ attentional control theory ideas. The suggested idea was that instructions are understood better when given in an auditory format as the information is parsed separately to visual information. This is tested in parallel with just-in-time information, with separate tutorials that offer either voice recorded hints or text pop-up boxes. (White, 2014) claims that only one of audio or text should be offered for tips as opposed to both as this could lead to cognitive overload – hence using both will also be tested to verify this claim.

The final idea tested in this project is Cognitive Apprenticeship with the idea that when a user views an experienced player completing tasks they are more likely to improve themselves – as defined by (Chang et al., 2013). This tutorialisation method is chosen to gain a better understanding of how easy it is to implement apprenticeship methods as well as to see their effectiveness in improving the user's play – as well as if it detracts from any flow experience.

3.8 Implementing Just in Time Instruction

To produce the tutorials concerning just-in-time instruction first the control version was created as shown below:

Figure 10: The online manual for the Text at Start tutorial

It was aimed to still make use of all recommendations from the background research hence images and text were used to follow the imagens and logogens ideas of dual coding theory. The player will be shown this guide before they play through the tutorial however it will not be accessible throughout the rest of a playthrough. This highly visual approach aims to make the mechanics more memorable however it is predicted this will still be less effective than just-in-time instruction.

Just-in-time instruction is defined by being able to identify the moment where a player is struggling and then being able to give them a contextualised hint as to what they should do next. Hence there are two key parts to Just-in-time: the identification and the response. An overall tutorial manager class was created which contained an array of the tips for the project. It also maintained a boolean array to check if a tip had already been shown to make sure the player does not receive tips on repeat attempts of the level- meaning they are less likely to be frustrated in-game. The one function of this class was ensuring that tips were outputted and this was done through the Update function.

3.8.1 Implementing Tutorial Tips

Here the responses to an issue are identified. By following through each level of the tutorial and considering it's connected mechanics a series of 17 tips were identified that could appear throughout the tutorial to offer guidance. A tip game object was created that stored a simple UI pop-up and an audio source for both the delivery options of a tip. A custom script was also included that contained the tip text, the audio file for the tip, and a duration for when a tip should appear. The tip contains one essential method named showTip which when called will output the tip in the format designated by an input parameter specifying the delivery type.

The tip offers three delivery types: text, audio, or both as specified in 3.7. In designing the text/speech for a tip, care was taken to ensure a conversational approach as recommended by the personalisation principle (Mayer, 2009). The audio was not outsourced due to the limitations of this project and was recorded and edited in audacity. Appendix B shows a representation of each tip by index number in the tip array stored by the game's tutorial manager class.

3.8.2 Implementing Triggers

The other key part of just-in-time instruction is the identification and this is done via a series of triggers and timers. Triggers are simple collision detection boxes within a game world that create some response when a specific object collides with them. In this project, triggers were implemented as collision boxes with a script containing a Boolean flag that was raised when a player first collided with it. Timers were made up of two components the timer variable itself and the max value at which a response would be triggered.

Individually, an example of where a trigger was used is in the first level – if the player hit the spikes then the appropriate hint was called. As an example for a timer by itself, also in the first level, if the player is detected not to have moved in the first 2 seconds of the game then the appropriate tip is given. Via combinations of triggers and timers more complex detection of whether a player is struggling can take place.

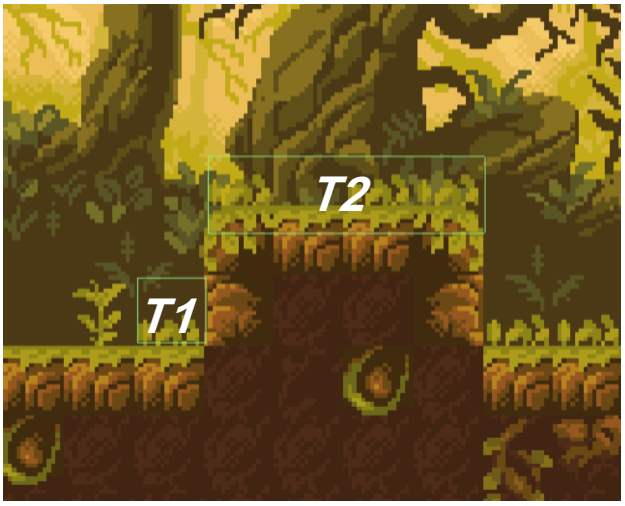


Figure 11: Example of the use of multiple triggers to detect player's struggling

For example, again in the first level, two triggers and a timer are made use of for giving the jumping tip. In this case, a timer begins counting when a trigger at the base of the raised platform (*T1*) is collided with. If the player hits the second trigger (*T2*) on top of the platform the timer is halted, however, if the max value is reached before this happens the jumping tip is given. Another style of trigger used is by counting player deaths. In level 8 this is made use of to give a tip about respawning in the longer level if the player has died more than a certain amount of times however, this tip only makes sense if the player has passed a certain point in the level so a trigger is added to ensure that this counter is only increased if the trigger's flag has been raised. Appendix C offers a full and comprehensive list of all the triggers and their usages throughout this project.

Once all the triggers were in place, each level had a custom script applied that added in the necessary logic to interact with the tutorial manager and give out hints when the corresponding triggering conditions had been met. In this way just in time instruction was completed – with no harder difficulty incurred from using audio as opposed to text – apart from potentially the quality of delivery due to the limitation of not being able to use professional-quality recordings.

3.9 Implementing Cognitive Apprenticeship

As discussed in the background research (2.8) cognitive apprenticeship in games is based on a set of principles defined by (Chang et al., 2013). Here the focus of the implementation is demonstration with ideas such as scaffolding and training already integrated. (White, 2014) recommends using an NPC character for demonstration and this idea will be used in the game – with the idea that the

player is chasing an NPC through the levels with this NPC demonstrating examples of how a level should be completed.

This idea offers two ideas for implementation. The first is to develop simple AI based on finite state machines- here the NPC would be adaptable and the player would be able to continue playing whilst the NPC also completes the level. The second solution is to record a video of an expert player completing the level and overlaying this at the start of each level to demonstrate to the player. This has the disadvantage of not allowing the player to play whilst the video is shown, however, it has the advantage of being computationally faster and much easier to implement- due to the timescale of this project this second solution was chosen.

Once the rest of the game was developed, example playthroughs of each level were recorded and stored in a streaming assets folder within the Unity project. This allowed them to be made use of within a browser. Next, a separate UI canvas was created to overlay these videos at the start of each level dependent on whether Cognitive Apprenticeship was turned on within that playthrough. Custom scripting detected the end of a video dynamically and then hid the UI canvas until the next level was reached and the next video was played.

3.10 Upload and Format for Testing

With all these stages complete the final task necessary was to format the selection of tutorials and provide a menu to access each one. Appendix D details the list of tutorials and the style of tutorial implemented in each one. 6 different tutorialisation methods were developed with a short and long variant for each resulting in a total of 12 tutorials. A menu was created which allowed a user to select tutorials and view the participant agreement, with UI buttons used to allow a user to easily select the options they wished to use. Basic usability principles were considered with the option to always return to the menu through a hotkey added (an idea from Nielsen's heuristics- that users must be able to back out of an action at any time (Nielsen, 1994)). The below image shows the tutorial selection screen. It was decided to number tutorials so as not to reveal the specifics of the difference between each tutorial as it was a goal that participants should be able to identify these differences for themselves during a research session.

Figure 12: The Tutorial Selection menu

Once the menu had been made, the buttons were linked to the tutorial manager, whereupon tutorial selection, booleans would be set to define the tutorial type to be given- this was maintained across scenes making use of the singleton pattern recommended for Unity design. Once all tutorials were connected up the project was built and loaded into a git repository with an associated page ready for use in research.

3.11 Summary

This section on implementing tutorials ensured that the second objective of this project was completed. First, a base game was developed ensuring a complex set of mechanics was included to make sure that there were enough different ideas for a user to learn, then levels were designed following ideas such as skill gating and designing for affordances. Then a series of tutorialising methods were chosen and implemented to develop a series of 12 tutorials. It was seen how easy methods could be used to include more developed tutorials – with one key note being that audio methods were just as easy as text methods. Cognitive Apprenticeship was developed in a fast manner more similar to a prototype than final software however this is believed to be sufficient for this project. The project is accessible at (Taylor, 2021) and will be made use of in the user research section of this project.

4. Testing & Usability Research

4.1 Introduction

Within this section, user research will be completed making use of the tutorials developed in the previous section. First, a brief explanation of the testing of software will be fulfilled before an explanation of how usability research was completed by making use of the think-aloud protocol. Then an analysis of these results will be completed making use of the affinity mapping process to identify key ideas from the qualitative research.

4.2 Software Testing

Within this project, due to the Agile development method, testing happened frequently – with focused testing happening at the end of each sprint. Black box testing was conducted making use of the sprint goals as tests to check what had been achieved and what would need fixing in the next sprint. The git repository was updated weekly to ensure checking was done on the final delivery platform of a web browser, which was able to ensure fewer errors were found in the final software. One example of this was with the cognitive apprenticeship videos which appeared in the Unity environment however not on the web as uncovered during functional testing. Due to this useful uncovering of an error, it was fixed for the final version of the game.

Due to the understanding that participants may access the tutorials from various web browsers, environmental testing was done on the four current most popular web browsers: Chrome, Safari, Edge, and Firefox (Wikipedia, 2021). This testing found that all four browsers could play the game at a sufficient level with a slight performance boost seen within Google Chrome.

Via the iterative black box testing done throughout the project and this final environmental testing, assurances were made that the final game tutorials fulfilled the specification outlined throughout the design and implementation section.

4.3 Usability Testing

Any type of testing conducted on users is known as usability testing. To gain a good understanding of what potential users thought of different styles of tutorial it was clear that this type of testing needed to be carried out. In (Andersen et al., 2012), a quantitative approach was carried out in which time played was the main factor of measurement, one of the recommendations for further work given is to attempt a more qualitative study to understand users feelings on different tutorial types and hence that is one of the goals here.

4.3.1 Think Aloud Protocol

The methodology of usability testing chosen for this project is the think-aloud protocol recommended by (Nielsen, 2012). The idea here is to ask the participant to constantly speak aloud their thoughts and feelings whilst conducting tasks set by the research facilitator. The session will be recorded and notes can be made on interesting points the participant makes. Think aloud offers robust data due to the fact participant answers are not guided and hence less likely to be biased and is also quick and easy to conduct. Research carried out by (Theodorou, 2010) identifies that concurrent think-aloud is the best practice for use in video game testing – despite the addition to cognitive load- he claims that the increase in load is minimal and tended not to affect the results.

4.3.2 Planning the Sessions

Knowing that think-aloud could be used to facilitate understanding of a participant's thoughts, a document was prepared to define how a research session would take place, this can be seen in

Appendix E. It was decided in agreement with (Dworkin, 2012) 5 participants will be chosen. In each session, the participant will play through a selection of the tutorials as decided by the facilitator throughout the session adapting to the participant's responses. The goal will be to expose the participants to each of the tutorial types and understand comparatively which of the methods they find most useful and enjoyable. The facilitator will aim to make notes throughout the session following up on moments from thinking aloud in between each tutorial attempt – this will allow a richer understanding of what a participant's opinions are.

4.3.3 Tools Used

Microsoft Teams - It was decided that to gain the most accurate collection of data possible session recording should take place including screen capture. Due to current restrictions, the only format testing could take was online and hence some form of video communication was also necessary. The tool that fulfilled these requirements was Microsoft Teams. This allowed the facilitator to organise the sessions through video calls within the service and record the sessions making use of the Microsoft Stream functionality. As Teams allowed screen sharing this could also be captured allowing the fullest collection of data possible.

Affinity Mapping – This tool was chosen as the method by which to analyse the results of the qualitative study. It is an extension and visualisation technique for thematic analysis that makes use of post-it notes to collate ideas and find themes. As specified in (Pernice, 2018) affinity mapping is a two-stage process – first post-it notes are generated by writing key points from usability sessions on to them, then in the second stage these notes are organised into groups based on concordant themes. One recommendation that will be used is colour coding for different participants to easily identify where data came from.

4.4 Results & Analysis

Over a week, five research sessions were carried out following the parameters laid out in Appendix E. Participants were recruited via social media advertisement with younger people (Ages 18-23) being the only respondents. From this, 5 participants were chosen and remote usability testing was carried out with the average session lasting 40 minutes. In a preliminary interview, each participant had basic information recorded including age, gender, affinity with video games, and the participant’s learner type. This provided a basis to develop deductions on the difference in experience within the participants as well as to understand why certain tutorialisation methods worked better for some participants. Appendix F represents the information of all participants anonymously.

Overall, all sessions ran smoothly with one reoccurring issue being, participants were reluctant to stop one tutorial and try another with multiple stating “Just one more go”- this could indicate multiple ideas – one being the game was made too hard, and another being too much focus was put on creating a good more addicting flow experience. It is believed both can be true with the challenge level being high both a positive and negative in this case as all players reported high enjoyment of the game itself however it was found that with some participants it increased the length of a research session almost two-fold.

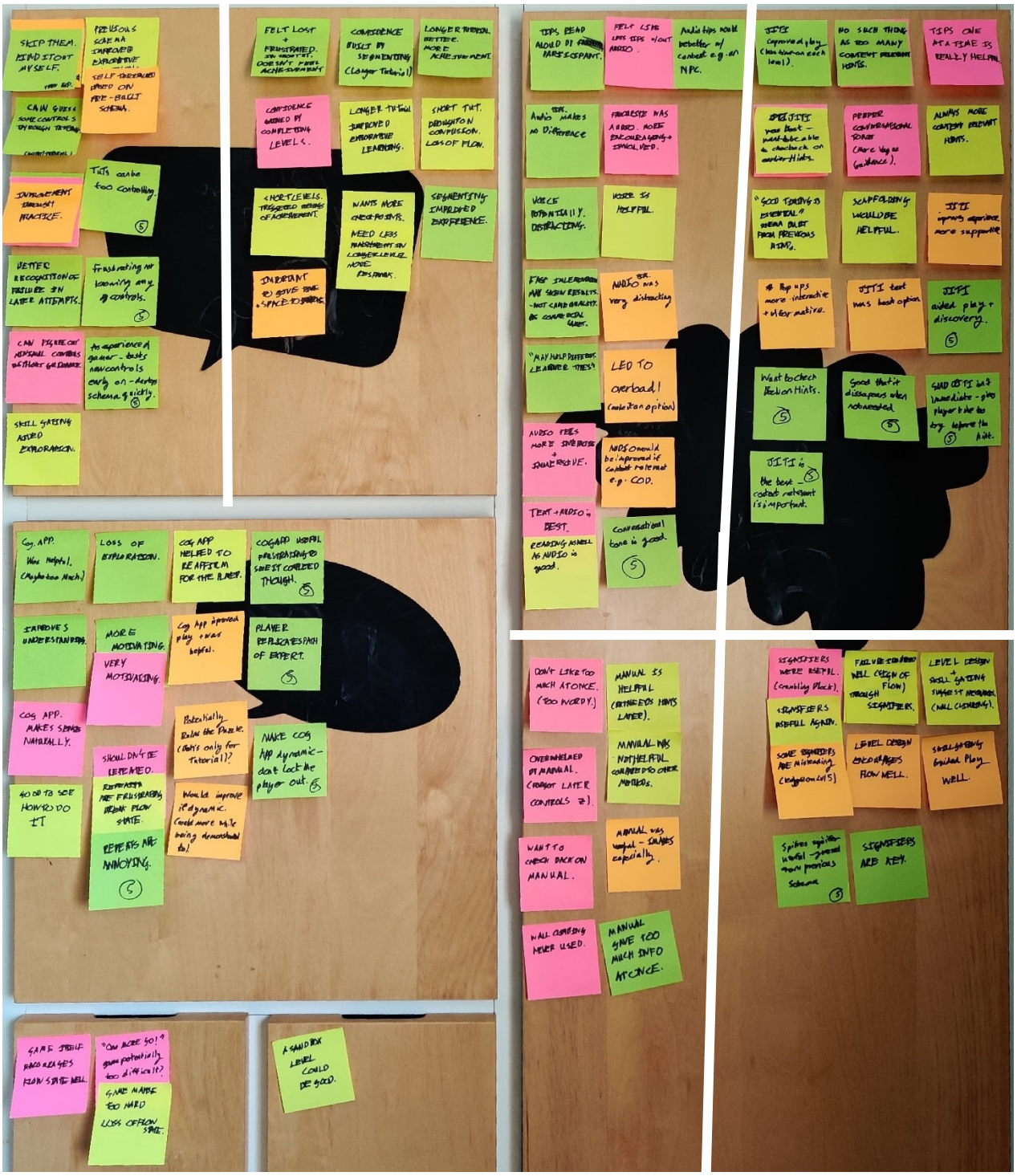


Figure 13: The Affinity Mapping Diagram

To analyse the data, affinity mapping was completed as shown in figure 12. The post-its were filled with key points collected from notes during sessions and in rewatches of the recordings of each session and then organised into sections as seen above focusing on different themes found in the research. A detailed image of each section can be found in Appendix G. Below detailed are the findings and ideas concluded from the affinity mapping.

One theme uncovered was in exploratory learning – with comments in preliminary questions stating that one participant found many tutorials too controlling – suggesting an improvement that could be made. The most exploratory of the tutorials were the first two which gave no didactic guidance. It was found that participants relied on past schema from other games to work out a couple of the controls with more experienced players picking this up faster and moving further through the

levels. Allowing play to be uninterrupted for experienced players was useful here, however, all players eventually found themselves stuck without any guidance whatsoever – with many stating they felt frustrated. Also regarding exploratory learning, it was seen in all participants that through practice (exploring the levels) all players improved dramatically throughout the tutorial, for example when a participant repeated a level they had completed before with a large number of attempts on their next playthrough this always took less time. This can also be down to good identification of success/ failure as part of improvement – an identifier for flow states.

One key element of getting this feedback in the game was the use of signifiers, which during think-aloud were commented on by all participants. As mentioned in the background research use of signifiers such as in this game, spikes or cracks on crumbling platforms players were able to identify the best way to navigate levels and understand the game on a higher level. This visual style of identification showed the quickest understanding of any mechanics although this may be due to the more simplistic nature of the environmental mechanics used. However, a point can be made that signifiers and design of art in this way may be a tool for game design, in general, the point that visual feedback improved learning is important to understand.

Also, key to flow and ensuring teaching was done efficiently was the level design and specifically skill gating. All participants were able to improve more quickly within the game setting as when they were taught mechanics they were then able to implement them at the same time in a subtle form of testing. The experienced gamers within the group were also able to identify the mechanics verbally that should be used to complete puzzles through the skill-gated designs that were used. This component of level design was key in allowing the participants to feel as if they explored whilst still checking that different skills had been learned and provided good checkpoints for triggers with just-in-time instruction also.

Another key idea tested was the use of a short vs long tutorial which provided a better understanding of Mayer's pretraining and segmenting principles as well as White's suggestions on punishment in tutorials. It was found that all participants found the longer tutorial which taught each mechanic on a separate level more useful than the short version that put them all in at once. Multiple participants felt a greater sense of accomplishment from completing the shorter sections of a longer tutorial and also found it helped them build confidence for the final level. It also gave more time to develop skills a key part of explorative learning again. Hence it is believed that the segmenting and pretraining principles are useful suggestions for tutorial design.

The shorter tutorial led to feelings of frustration and loss of flow when attempted before the longer tutorial, with one key issue being the lack of checkpoints along a larger level. This agrees with White's point on making punishments smaller within the tutorial which he suggested and was seen to increase confidence in players.

The first tutorialisation method given to participants that offered information on mechanics was the game manual present in tutorials three and four. Participants stated they found it useful to be able to see all the controls however once other methods were tried the majority of participants stated this was their least favourite way to have been taught. This was proven particularly with participant 2 who struggled to remember all the controls as presented (mostly the wall climbing) and hence struggling to complete the levels compared to other participants. Other participants stated it was overwhelming to be given all the information at one time – agreeing with cognitive load theory that too much information at once will lead to some information being ignored.

In contrast to the manuals, just-in-time instruction was the most popular method among all participants. It was stated that the game felt more interactive and immersive, agreeing with background research that believed just-in-time instruction would help maintain a flow state due to only offering help when needed. One worry was that just in time instruction may be overwhelming

however no participants found this with participants three and five, in particular, asking for even more context-relevant hints if possible. Participant five as an experienced gamer enjoyed the fact that they were not offered hints all the time and were given time to prove skills if they already knew how to use them and hence were not offered more instruction. One criticism given was that players could not check back on hints they had already been given and with an understanding of usability heuristics, this idea that you can go back to see what was previously said makes sense and the suggestion to allow a format to view past tips will be included in the framework. In line with this, one participant also asked for scaffolding: where tips would come back if a player continues to struggle and as per research from (Tsai et al., 2013), this idea will also be considered from the framework. Despite these two small criticisms just-in-time instruction was still favoured by all participants, with this method being the best & most supportive to improve the tutorial experience.

An area where participants' opinion was split was with the use of audio tutorialisation as suggested in dual coding and attentional control theory. Some participants felt it improved the experience with them feeling more encouraged by having a voice to guide them. These participants found the audio made the game more immersive with participant 2, an auditory learner, stating there were fewer hints when there was only text shown – although the same amount were present in both text and audio options. This may be due to the ideas of separate processing for audio put forward in attentional control theory that believe more attention is gained when multiple senses are used. In contrast for some participants, the voice was found to be distracting, although this was only seen in participants who weren't auditory learners.

Both participants who found it distracting also suggested that the implementation in this setting may also have been the issue with one of them suggesting Call Of Duty as a game in which they had experienced audio tutorialisation and enjoyed it with the important thing being that it felt more integrated into the game and more fitting to context. This agrees with a statement from (White, 2014) that any methods used should not break the immersion of the game and should make sense in context. One final suggestion to improve it was to offer the option to have audio – with the idea that customizability would improve tutorials and with the evidence seen through the research that all players had varying experiences it's clear that providing options would give the most useful approach. Overall, concerning this method, the consensus was that the combination of both audio and text popups simultaneously was believed to be best, despite the researcher's hypothesis that this could have led to cognitive overload due to the higher processing space needed within the mind.

The personalisation principle was made use of for both the text and audio pop-ups used in the game with multiple participants commenting on the format hints took. Participants enjoyed the more conversational tone hints took and disliked more didactic approaches that were sometimes taken. It was also seen that vaguer hints that guided the player as opposed to telling them the answers were also preferred which again ties back to exploratory learning.

The use of demonstration as suggested by literature on cognitive apprenticeship was found to be helpful to all participants – with participants tending to replicate the routes through levels that the videos showed them if they weren't following those methods before. Participant 3 stated that it was useful in reaffirming ideas they had and useful in providing guidance. The majority of participants found demonstration motivating particularly among visual learners, however, participant 5 found it infuriating to see the level completed when they could not finish it. Some commented that it potentially ruined the puzzle of each level however in a full game this would only be within the tutorial hence meaning it does not ruin the game itself. Another complaint was that the video repeated every time the player respawned meaning the participant was forced to watch it again which again broke immersion and flow leading to frustration. This was expected by the researcher and further proves the point that players dislike forced tutorials and not being able to try things

themselves. Hence, the solution recommended was to make demonstration more dynamic, where the player could continue to play while the demonstration happened. This would also allow replication to happen more easily as the expert could be easily followed. It was suggested that this approach was something participant 5 was comfortable with from other games and thought would be useful. They also recommended that the demonstration should have more game context similar to the recommendation for audio tutorialisation. Despite these criticisms, participants overall found cognitive apprenticeship to be a useful method in learning how to play and particularly in how to navigate difficult sections of levels, suggesting it may also be highly useful in open-world games where the player can become easily lost.

4.4.1 Results Summary

To summarise the findings, the most useful method for tutorialisation found was the use of just-in-time instruction, with users appreciating how it only gave information when necessary and at the moment when it was needed. The two further recommendations for this would be even more context-relevant hints and some way to check back on the hints given earlier in the tutorial.

Another key component of tutorialisation that worked well was giving as much space and time for exploratory learning as possible. Players get a sense of achievement from figuring things out for themselves, but guidance should still be used with one key recommendation for this being the use of dynamic skill gating in the level design. To keep that feeling achievement high and to also not overwhelm the player it is also recommended that segment should be used to teach each mechanic one at a time, this built confidence in all participants.

Finally, in regards to audio tutorialisation and cognitive apprenticeship, it was found that the methods split opinion more but could be improved by giving them more context. People appreciated the use of audio for immersion but would prefer it if it came from an NPC, and was also found to be most useful when accompanied by text pop-ups. When it came to cognitive apprenticeship the issue was more related to the forcing of players to watch demonstrations repeatedly and this could be improved by making it more dynamic and allowing the player to continue playing alongside it.

5. Evaluation & Conclusion

5.1 The Proposed framework

Figure 14: The Proposed Framework for Tutorial Design

Based upon the research completed using surrounding literature, commercial games, and usability testing the above framework is suggested as a set of ideas for improving tutorial design. Three key sections are identified with key points within each that can improve how tutorials are given. Most ideas are heavily influenced by the results of the usability testing with just-in-time information being one prevalent idea proven from there. Skill gating is an idea again proven in testing but also proven to be useful in the implementation as a way to implement other ideas including the triggers for instruction design. Some ideas particularly in the integrated ideas section take more inspiration from background literature as it is believed the implementation used for testing was subpar and hence influenced results. This is specific to audio and Cognitive Apprenticeship, where research participants described good examples in commercial games but found the version in testing less useful.

5.2 Achievement of Aim & Objectives

To understand if the aim has been accomplished fully, each objective can be analysed against the work done to gain an understanding of how well the aim was completed.

The first objective was to identify at least 5 methods for tutorialisation, with this being explored in the background research section of this dissertation. A strong understanding of psychological theories around video games was undertaken, for example flow states, to be able to give reasoning to different methods. Then methods such as Mayer's principles and just-in-time instruction were explored for methods that could be implemented and tested.

The second objective of developing at least 3 tutorials was easily accomplished with a set of 12 tutorials that provided exploration on different ideas from the literature and the implementation is described in the third section of this dissertation. The 2D platformer chosen was a good choice as a basis for the game from a development standpoint as there were a multitude of mechanics that could be introduced and many ways to provide tutorialisation. However, with a more complex genre, better implementation of cognitive apprenticeship and dual coding could have been produced in this area. There were also a couple of small errors in the final tutorials – with moving back to previous levels forcing you to re-complete them being the main issue. Overall, this objective was completed well due to exceeding the boundary of 3 tutorials however it is clear that steps could have been taken to improve the experience further.

The third objective was to conduct user research on at least 5 participants with this objective also being fully achieved. The completion of this objective is described in the testing section of this dissertation and with the aid of the tutorials developed, 5 participants were tested on the usability of different tutorials making use of the think-aloud protocol. Again, this objective was a success with a possible improvement being including more participants.

The fourth objective was to analyse the research and make use of the results to produce the framework. The analysis was achieved in the second part of the testing section, making use of the affinity mapping technique to organise qualitative data and compare it with ideas from the background literature to see if testing had been a success. The themes developed here were then used to produce the framework seen at the start of this section completing this objective fully.

The overall aim of this dissertation was to produce a framework for well-designed video game tutorials and it is clear that this has been achieved through the objectives. Ideas from each of the 3 main sections of background research, implementation, and testing were able to contribute to creating a framework that offers up 12 suggestions to improve tutorialisation as expected in the aim.

5.3 What Was Learnt

Throughout this project and dissertation, I have worked to understand many new ideas regarding game design, development, and theories. One key area of understanding was how multiple fields such as education and psychology can be used to provide theoretical reasoning for the inclusion of specific ideas to design products. This is an important aspect of HCI that has been understood through this project. Another new area explored was in usability testing, where new techniques such as think-aloud and affinity mapping were attempted for the first time. I believe both of these new methods were a success and can be applied to future projects as useful tools.

Concerning game development, new skills in working with a game engine were developed, as well as C# scripting skills. Skills regarding design and implementation including levels, mechanics, and

UI were also used and improved through background reading to understand how to provide the best user experience.

New knowledge into tutorialisation in games was the focus of this project and ideas surrounding this were developed massively, with problem-solving skills used to introduce solutions to cognitive apprenticeship and just-in-time that made use of triggers and timers. Critical thinking skills were developed in analysis being able to rely on the new knowledge on methods to evaluate whether the tutorials developed achieved the objectives of finding the most helpful and fun solutions to the problem of tutorialisation.

5.4 Improvements & Further Work

In regards to improvements within this project, one issue found was with recruiting a sufficient amount of participants for research, with this potentially due to mitigating circumstances. Further work could be done with a larger scale study exploring how well the tutorials worked on more people, with one specific area being a wider age range with this research using participants in a younger age range.

In regards to the game itself, one issue found was the game was too addictive distracting the participants from the tutorialisation. If this was a commercial game this would be a success however with the focus on tutorials an improvement would be to lower the challenge level of the game making it easier to focus on the tutorials. Another improvement here would be to add more polish to the game, with more animation and a larger level set.

Further work could also be done to explore the longer-term effects of different tutorialisation types where improvement of participants could be tested when only using specific types of tutorial with a form of AB testing. Other studies could also be undertaken into differing game genres to understand if these differing genres would affect the tutorialisation methods that should be used.

Finally, the fast implementations of audio tutorialisation and cognitive apprenticeship in this project resulted in differing results to the literature, and hence more study could be undertaken into better methods to introduce these techniques to tutorials. Cognitive apprenticeship in particular could be explored via the use of neural networks to develop NPC’s that can demonstrate and offer instruction better to the player in times of need.

References

A Perfect Jump in Unity - A Complete Guide. Youtube.com. (2019). Retrieved 16 April 2021, from https://www.youtube.com/watch?v=RPdn3r_tqcM.

Activision. (2020). Call Of Duty: Warzone [Multiple].

Alcorn, A. (1972). Pong [Arcade]. Atari.

Andersen, E., O'Rourke, E., Liu, Y., Snider, R., Lowdermilk, J., Truong, D., & Zoran Popovic, S. (2012). The Impact of Tutorials on Games of Varying Complexity. CHI '12: Proceedings Of The SIGCHI Conference On Human Factors In Computing Systems.

Atlassian. (2021). Trello [Browser/ Mobile].

Audacity Team. (2021). Audacity [Multiple]. The Audacity Team.

Bardzell, J., Bardzell, S., & Ryan, W. (2020). The Video Game Tutorial: Narrative, HCI, and Virtual Learning. Retrieved 25 March 2021, from https://studylib.net/doc/8939088/the-video-game-tutorial--narrative--hci--and-virtual-lear...

Bethesda Softworks. (2008). Fallout 3 [Multiple].

Cai, H., & Lin, Y. (2011). Modeling of operators' emotion and task performance in a virtual driving environment. International Journal Of Human-Computer Studies, 69(9), 571-586. https://doi.org/10.1016/j.ijhcs.2011.05.003

Cattell, R. (1971). Abilities: Their structure, growth, and action. Houghton Mifflin.

Chang, W., Chang, C., Yang, H., & Jheng, M. (2013). Game-Based History Ubiquitous Learning Environment Through Cognitive Apprenticeship. Information Technology Convergence, 617-625. Retrieved 7 April 2021, from https://link-springer-com.libproxy.ncl.ac.uk/book/10.1007%2F978-94-007-6996-0?page=2#toc.

Clark, J., & Paivio, A. (1991). Dual coding theory and education. Educational Psychology Review, 3(3), 149-210. https://doi.org/10.1007/bf01320076

Collins, A., Brown, J., & Newman, S. (1988). Cognitive Apprenticeship. Thinking: The Journal Of Philosophy For Children, 8(1), 2-10. https://doi.org/10.5840/thinking19888129

Creative Commons. (2021). Creative Commons — Attribution 2.0 Generic — CC BY 2.0. Creativecommons.org. Retrieved 16 April 2021, from https://creativecommons.org/licenses/by/2.0/.

Csikszentmihalyi, M. (1990). Flow: The Psychology of Optimal Experience. Joosr Ltd.

Dillon, R. (2010). The 6-11 Framework: a New Methodology for Game Analysis and Design. Retrieved 7 April 2021, from.

Dworkin, S. (2012). Sample Size Policy for Qualitative Studies Using In-Depth Interviews. Archives Of Sexual Behavior, 41(6), 1319-1320. https://doi.org/10.1007/s10508-012-0016-6

Extremely OK Games. (2018). Celeste [Multiple].

Forced Tutorial - TV Tropes. TV Tropes. (2021). Retrieved 1 April 2021, from https://tvtropes.org/pmwiki/pmwiki.php/Main/ForcedTutorial.

Gallant, M. (2009). Mechanics, Dynamics & Aesthetics - The Quixotic Engineer. Gangles.ca. Retrieved 7 April 2021, from http://gangles.ca/2009/08/21/mda/.

Gee, J. (2005). Learning by Design: Good Video Games as Learning Machines. E-Learning And Digital Media, 2(1), 5-16. https://doi.org/10.2304/elea.2005.2.1.5

Gozli, D., Bavelier, D., & Pratt, J. (2014). The effect of action video game playing on sensorimotor learning: Evidence from a movement tracking task. Human Movement Science, 38, 152-162. https://doi.org/10.1016/j.humov.2014.09.004

Guerrilla Games. (2017). Horizon Zero Dawn [PlayStation 4].

Hicks, K., Dickinson, P., Holopainen, J., & Gerling, K. (2018). Good Game Feel: An Empirically Grounded Framework for Juicy Design. Proceedings Of Digra. Retrieved 7 April 2021, from.

Hunicke, R., LeBlanc, M., & Zubek, R. (2001). MDA: A Formal Approach to Game Design and Game Research. Retrieved 7 April 2021, from.

Interaction Design Foundation. (2021). The Encyclopedia of Human-Computer Interaction, 2nd Ed. (p. Chapter 16).

Iyer, A. (2017). Analyzing Super Mario's level and tutorial design. Medium. Retrieved 1 April 2021, from https://medium.com/swlh/the-perfect-game-tutorial-analyzing-super-marios-level-design-92f08c28bdf7.

Joiner, B. (2018). How To Get Started With Scrum and Trello For Your Team At Work. Blog.trello.com. Retrieved 13 April 2021, from https://blog.trello.com/how-to-scrum-and-trello-for-teams-at-work.

Kaptelinin, V. (2021). Activity Theory. The Interaction Design Foundation. Retrieved 1 April 2021, from https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-computer-interaction-2nd-ed/activity-theory.

Keith, C. (2010). Agile Game Development with Scrum (1st ed.). Pearson Technology Group Canada.

Kimball, S., & Mattis, P. (2021). GIMP [Multiple]. GIMP Team.

Kirschner, P., Sweller, J., & Clark, R. (2006). Why Minimal Guidance During Instruction Does Not Work: An Analysis of the Failure of Constructivist, Discovery, Problem-Based, Experiential, and Inquiry-Based Teaching. Educational Psychologist, 41(2), 75-86. https://doi.org/10.1207/s15326985ep4102_1

LaCapria, K. (2019). Was the Original Super Mario Bros Only 40 KB (While Modern.... Truth or Fiction?. Retrieved 1 April 2021, from https://www.truthorfiction.com/was-the-original-super-mario-bros-only-40-kb-while-modern-screenshots-of-one-frame-are-283-kb/.

Laraman, M. (2018). The ultimate guide to video game tutorial design. MikeLaraman.com. Retrieved 6 April 2021, from https://mikelaraman.com/the-ultimate-guide-to-videogame-tutorial-design.

List of best-selling video games - Wikipedia. En.wikipedia.org. (2021). Retrieved 1 April 2021, from https://en.wikipedia.org/wiki/List_of_best-selling_video_games.

Lohse, K., Shirzad, N., Verster, A., Hodges, N., & Van der Loos, H. (2013). Video Games and Rehabilitation. Journal Of Neurologic Physical Therapy, 37(4), 166-175. https://doi.org/10.1097/npt.0000000000000017

Mann, B. (2008). The evolution of multimedia sound. Computers & Education, 50(4), 1157-1173. https://doi.org/10.1016/j.compedu.2006.11.002

Mann, B., Newhouse, P., Pagram, J., Campbell, A., & Schulz, H. (2002). A comparison of temporal speech and text cueing in educational multimedia. Journal Of Computer Assisted Learning, 18(3), 296-308. https://doi.org/10.1046/j.0266-4909.2002.00241.x

Mann, B., Schulz, H., & Cui, J. (2012). Speech Cueing on the Web by 'The Little Dude'. International Journal Of Online Pedagogy And Course Design, 2(3), 32-44. https://doi.org/10.4018/ijopcd.2012070103

Mayer, R. (2009). Multimedia learning (2nd ed.). Cambridge University Press.

Metzger. (2010). The Escapist : Forums : Gaming Discussion : I Hate Tutorials in Games!. The Escapist. Retrieved 1 April 2021, from https://v1.escapistmagazine.com/forums/read/9.397242-I-Hate-Tutorials-in-Games.

Miller, J. (2019). Visual Studio Tools for Unity. Docs.microsoft.com. Retrieved 16 April 2021, from https://docs.microsoft.com/en-us/visualstudio/gamedev/unity/get-started/visual-studio-tools-for-unity.

Moirn, R., Roberge, M., Léger, P., Senecal, S., Lefebvre, M., & Fredette, M. (2016). The Effect of Game Tutorial: A comparison between casual and hardcore gamers. CHI PLAY'16 Extended Abstracts, 229-237. Retrieved 26 March 2021, from https://dl.acm.org/doi/pdf/10.1145/2968120.2987730.

Mojang. (2011). Minecraft [Multiple].

Nacke, L., & Lindley, C. (2008). Flow and Immersion in First-Person Shooters: Measuring the player's gameplay experience. Proceedings Of The 2008 Conference On Future Play: Research, Play, Share, Future Play 2008. Retrieved 1 April 2021, from https://www.researchgate.net/publication/221643959_Flow_and_immersion_in_first-person_shooters_measuring_the_player%27s_gameplay_experience.

Nielsen, J. (2012). Thinking Aloud: The #1 Usability Tool. Nielsen Norman Group. Retrieved 20 April 2021, from https://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool/#:~:text=Definition%3A%20In%20a%20thinking%20aloud,keep%20up%20a%20running%20monologue.

Nielsen, J. (1994). 10 Usability Heuristics for User Interface Design. Nielsen Norman Group. Retrieved 18 April 2021, from https://www.nngroup.com/articles/ten-usability-heuristics/.

Nijman, J. (2013). The art of screenshake [Video]. Retrieved 7 April 2021, from https://www.youtube.com/watch?v=AJdEqssNZ-U.

Nintendo. (1985). Super Mario Bros. [NES].

Pernice, K. (2018). Affinity Diagramming: Collaboratively Sort UX Findings & Design Ideas. Nielsen Norman Group. Retrieved 22 April 2021, from https://www.nngroup.com/articles/affinity-diagram/.

Ramey, J., Boren, T., & Cuddihy, E. (2006). Does Think Aloud Work? How Do We Know?. CHI '06 Extended Abstracts On Human Factors In Computing Systems, 45-48. Retrieved 25 March 2021, from.

Shaherfi Hadi, M. (2019). Super Mario Bros 1-1: Understanding Player Psychology – Crazy Atelier. Oss.adm.ntu.edu.sg. Retrieved 1 April 2021, from https://oss.adm.ntu.edu.sg/muhammad384/super-mario-bros-1-1-understanding-player-psychology/.

Starloop Studios. (2020). Best Agile Practices in Game Development | Starloop Studios Blog. Game Development Outsourcing Company, 3D Art Outsourcing, 3D Animation Outsourcing | Starloop Studios. Retrieved 13 April 2021, from https://starloopstudios.com/best-agile-practices-in-game-development/.

Statt, N. (2020). Call of Duty: Warzone now has more than 50 million players one month after launch. The Verge. Retrieved 1 April 2021, from https://www.theverge.com/2020/4/10/21216683/call-of-duty-cod-warzone-players-milestone-50-million-activision.

Stout, M. (2015). How to train players right , so they don't hate learning to play. Gamasutra.com. Retrieved 1 April 2021, from https://www.gamasutra.com/view/news/253028/How_to_train_players_right_so_they_dont_hate_learning_to_play.php.

Sweller, J. (1994). Cognitive load theory, learning difficulty, and instructional design. Learning And Instruction, 4(4), 295-312. https://doi.org/10.1016/0959-4752(94)90003-5

Sweller, J. (2011). Cognitive Load Theory and E-Learning. Sydney; School of Education, University of New South Wales. Retrieved 1 April 2021, from https://link.springer.com/chapter/10.1007/978-3-642-21869-9_3.

Swink, S. (2008). Game Feel: A Game Designer's Guide to Virtual Sensation (1st ed.). CRC Press.

Taylor, B. (2021). Dissertation Tutorials. Berttdev.github.io. Retrieved 19 April 2021, from https://berttdev.github.io/Dissertation-Prototype/.

Theodorou, E. (2010). Let the Gamers Do the Talking: A Comparative Study of Two Usability Testing Methods for Video Games (Masters). University College London.

Tsai, F., Kinzer, C., Hung, K., Chen, C., & Hsu, I. (2013). The importance and use of targeted content knowledge with scaffolding aid in educational simulation games. Interactive Learning Environments, 21(2), 116-128. https://doi.org/10.1080/10494820.2012.705852

Ubisoft. (2012). Assassin's Creed III [Multiple].

Unity Technologies. (2019). Unity [Windows].

Video Game Art & Tidbits. Twitter. (2018). Retrieved 1 April 2021, from https://twitter.com/vgartandtidbits/status/1018184822378909696.

White, M. (2012). New tutorials for digital games: game design meets instructional design (Thesis (Doctoral (PhD)). Memorial University of Newfoundland.

White, M. (2014). Learn to Play: Designing Tutorials for Video Games (1st ed.). CRC Press.

Wikipedia. (2021). Usage share of web browsers - Wikipedia. En.wikipedia.org. Retrieved 20 April 2021, from https://en.wikipedia.org/wiki/Usage_share_of_web_browsers.

Workfront. (2021). Agile Project Management - A Beginner's Guide | Workfront. Workfront.com. Retrieved 13 April 2021, from https://www.workfront.com/project-management/methodologies/agile.

Youtube: Call of Duty Warzone Tutorial. Youtube.com. (2021). Retrieved 1 April 2021, from https://www.youtube.com/watch?v=0FiW5guZgJY.

ZapSplat. ZapSplat - Download free sound effects. (2021). Retrieved 16 April 2021, from https://www.zapsplat.com/.

Appendix A: Level Designs

Level 1

Level 2

Level 3

Level 4

Level 5

Level 6

Level 7

Level 8 / Short Tutorial (Part 1)

Level 8 / Short Tutorial (Part 2)

Appendix B: Just-In-Time Instruction Hints

Tip No	Tip
0	You can use left and right arrow keys to move
1	You can press C to jump over obstacles
2	Ouch! Watch out spikes will kill you and you'll have to try again
3	If you press C while in mid-air you get a second Jump!
4	Don't Forget! Press C to jump over obstacles
5	That mushroom down there looks bouncy- it should give you a boost!
6	Try not to press Jump too early!
7	Be careful! those platforms don't look very stable
8	You've got to be fast those platforms will crumble away when you step on them!
9	If you Hold Z you can grab hold of the wall, then climb up
10	Use the up and down arrow keys to climb walls
11	Not sure you can get up that way!
12	If you jump while on a wall you'll jump off it (make sure you're not still grabbing it though)
13	These platforms move if you land on that one you should be able to get across
14	Ouch! Watch out those spikes are super dangerous
15	Looks like you Respawn right back at the start on this one!
16	Press C to Jump if you press again in mid-air you double Jump

Appendix C: Just-In-Time Instruction Triggers

Long Tutorial

Level	Mechanic	Trigger
1	Navigating Levels	If no player movement for 2 Seconds at the start offer the tip
	Horizontal Movement	
	Jumping	If player reaches the first obstacle have trigger, then start a timer, if the player does not reach the second trigger by max val then offer the tip
	Spikes	If death to spikes, then offer the tip
2	Double Jumping	If death to the second set of spikes offer the tip
	Reinforcing Jumping	If death to the first set of spikes offer the tip
3	Bounce Pads	If player reaches the edge have trigger, then start a timer, if player does not reach second trigger (on bouncepad) by max val then offer the tip
	Spikes on the ceiling	If death to spikes on ceiling offer the tip
4	Crumbling Platforms	On entry to level offer tip
		If death to spikes offer the more explcit tip
5	Wall Grabbing	If death to the second set of spikes offer the tip
	Wall Climbing	If wall grabbed but not moving after a second offer the tip
6	Wall Jumping	If death to spikes on left before trying wall Jumping offer the tip
		If Wall grabbing on right and hasn’t jumped 1 second after hitting the trigger then offer the tip
7	Moving Platforms (Standable)	If death to spikes at bottom of the level offer the tip
	Moving Platforms (Spikey)	If death to spikey moving platform offer the tip

8	Respawning Further Back	If respawn two times and hit trigger past halfway point of the level offer the tip
	Vertical Spikey Platforms	If death to the spikey platform offer the tip

Short Tutorial

Mechanic	Trigger
Horizontal Movement	If no player movement for 2 Seconds at start offer the tip
Jumping	If player reaches the first obstacle have trigger, then start a timer, if the player does not reach the second trigger by max val then offer the tip
Spikes	If you land in spikes offer the tip
Double Jumping	Offer at the first obstacle alongside the jumping tip
Wall Grabbing	If you reach the large wall and don't grab it within two seconds offer the tip
Wall Climbing	If wall grabbing tip has been given and not climbing within a second then offer the tip
Bounce pads	When arriving at the jump pad section offer the tip
Wall Jumping	If in spikes at this section offer the tip
Crumbling Platforms	If in spikes at this section offer the tip
Moving Platforms	If Hit spikes at this section offer the tip
Respawning at the Start	If respawn two times and hit trigger past halfway point of the level offer the tip

Appendix D – The Tutorial List

Tutorial Number	Short or Long	Tutorial Style
1	Long	No Tutorial
2	Short	No Tutorial
3	Long	Didactic Text at Start
4	Short	Didactic Text at Start
5	Long	Conversational Just in Time Text
6	Short	Conversational Just in Time Text
7	Long	Conversational Just in Time Audio
8	Short	Conversational Just in Time Audio
9	Long	Conversational Just in Time Audio + Text
10	Short	Conversational Just in Time Audio + Text
11	Long	Cognitive Apprenticeship with Conversational Just in Time Text
12	Short	Cognitive Apprenticeship with Conversational Just in Time Text

Appendix E: Designing User Research

Research Objectives

The research objective must coincide with the overall aims and objectives of this project. Focusing on these the following objective will be the focus of this usability testing: To understand the value of different methods of tutorialisation in the created video game. Value will be measured by two main qualitative measures: enjoyability (how much a participant has fun during a specific tutorial) and usefulness (how helpful the tutorial is in understanding how to complete a level)

Session Rules for Facilitator:

- First, pick randomly between tutorial 1 or 2 for each participant

- The loop for the session is as follows:

    - When participant gives up or completes tutorial ask questions relating to their thoughts during the tutorial.

    - Pick the next tutorial based on this

    - Return to the start of the loop

- Participants must complete one each of [1 or 2] and [3 or 4] and [5, 6, 7 or 8] and [10 or 11] and [11 or 12]

- Participants must be exposed to both short and long tutorials within their session.

Script for Research

First, do introductions and pleasantries. Then direct the participant to the website to access the tutorials. Next, direct them to the participation agreement and ask them to read through it. Then explain the purpose of the research as follows:

"This is a research session to understand the thoughts of users to understand the most enjoyable and useful methods for teaching you how to play a video game – in this case a 2d platformer [ask if they understand what a platformer is and if not then explain].

In this session, you will play through a series of different tutorials designed around getting a character to the end of the same set of levels. To understand your thoughts during this process we are going to use a process called the think-aloud protocol. This works as follows: This session will be recorded, and you will be asked during each playthrough to speak aloud your mind. This means no filtering of your thoughts just as soon as a thought, opinion or feeling comes to mind concerning the research simply tell us. The more you can talk the better and you might be reminded to keep going if you stop talking- but do not worry if you cannot think of anything to say just keep going and something will definitely come up.

As this is prototype software, we understand there might be errors in the gameplay however we ask that when possible please focus on the tutorial elements- but if something is bugging you please let us know! If you feel you cannot or do not want to complete a tutorial please let us know and we can stop and try another version. Finally, we ask that as you approach each tutorial you approach it as if you are learning the game for the first time, we understand this can be difficult but please try your best. Do you have any questions? [answer questions]"

At this point ask them to ensure sound is turned on, on their device and ask to begin recording and when given permission, start recording and ask for verbal agreement with the participation agreement. Once this is done give the participant a number, ask these questions and note down the answers:

What gender do you identify as?

What is your age?

Could you quickly complete these questions to identify your learner type?

(https://arden.ac.uk/what-type-learner-are-you)

On a scale of 0 to 5, how much experience do you have playing video games? 0 being none and 5 being playing video games on a regular (daily) basis

What types of video games do you enjoy playing?

(Ideas for follow-up: why do you like this type of game? How often do you play this game?)

What are your opinions on Video Game Tutorials?

(Ideas for follow-up: why is it that you like/ dislike them? Any example of a video game tutorial they may have enjoyed and why?)

Next, ask them to go into the tutorials and by following the session rules conduct the session. At the end of each tutorial, playthrough ask questions to elaborate on thoughts from their thinking aloud. Suggestions are as follows:

Did you find this tutorial better or worse than the last one? And why?

How could it be improved?

You mention "X" during this tutorial could you elaborate on what you mean?

You mentioned you felt "confused/ nervous/ focused/frustrated/etc." during this tutorial, why did you feel this way?

When the end is reached, follow up with a couple of post-play questions:

What were the methods that you found most enjoyable/ useful in teaching you how to play?

Why were they the most enjoyable/useful to you?

Were there any methods you found not enjoyable/ useful in teaching you how to play? If so what were they and why were they not useful?

Is there anything else that may have been useful to you in learning how to play this game?

Thank the participant and remind them recordings will be destroyed by the 7th May 2021. Then ask if they have any other questions and answer them if so then stop recording and end the session.

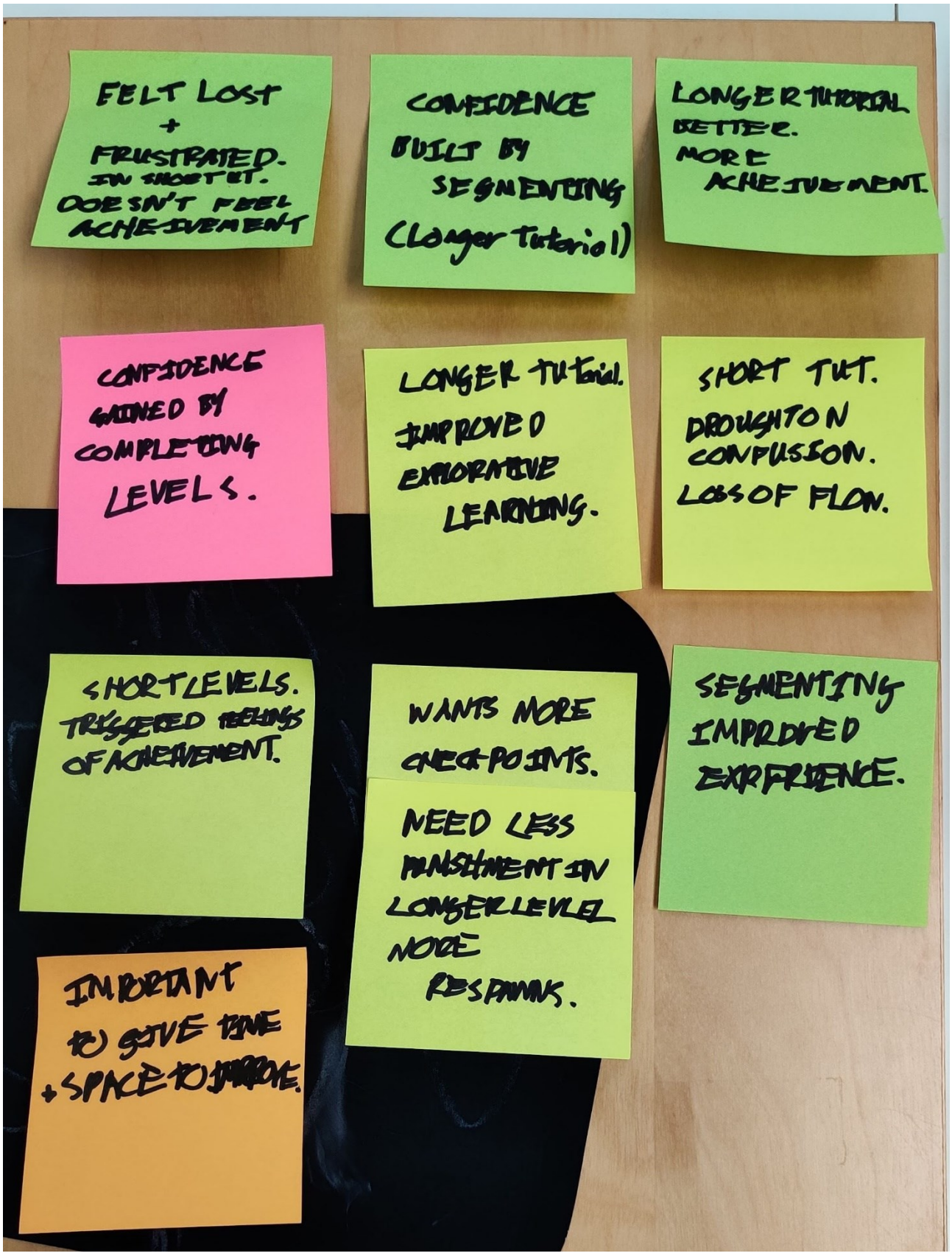
Appendix F: Participant Information

Participant No	Age	Gender	Experience Playing Video Games (0-5)	Learner Type	Session Length (minutes)	Post It Color
1	20	Female	3	Auditory/ Visual	25	Green
2	19	Female	2	Auditory	28	Pink
3	21	Female	4	Auditory/Kinesthetic	53	Yellow
4	21	Male	4	Kinesthetic/ Visual	59	Orange
5	21	Male	5	Visual	35	Green with 5 on it

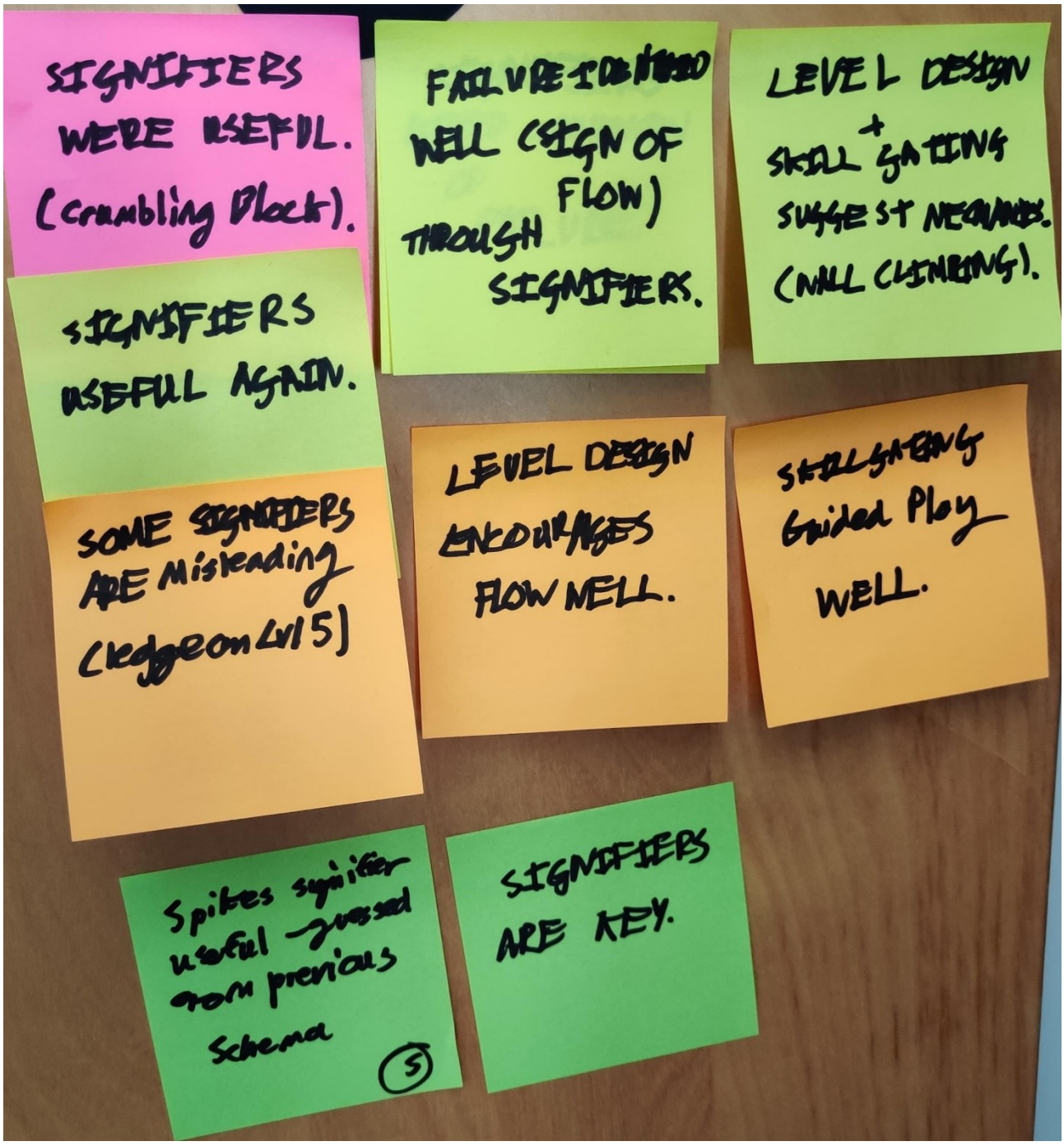
Appendix G: Affinity Mapping Images

1 - Exploratory Learning

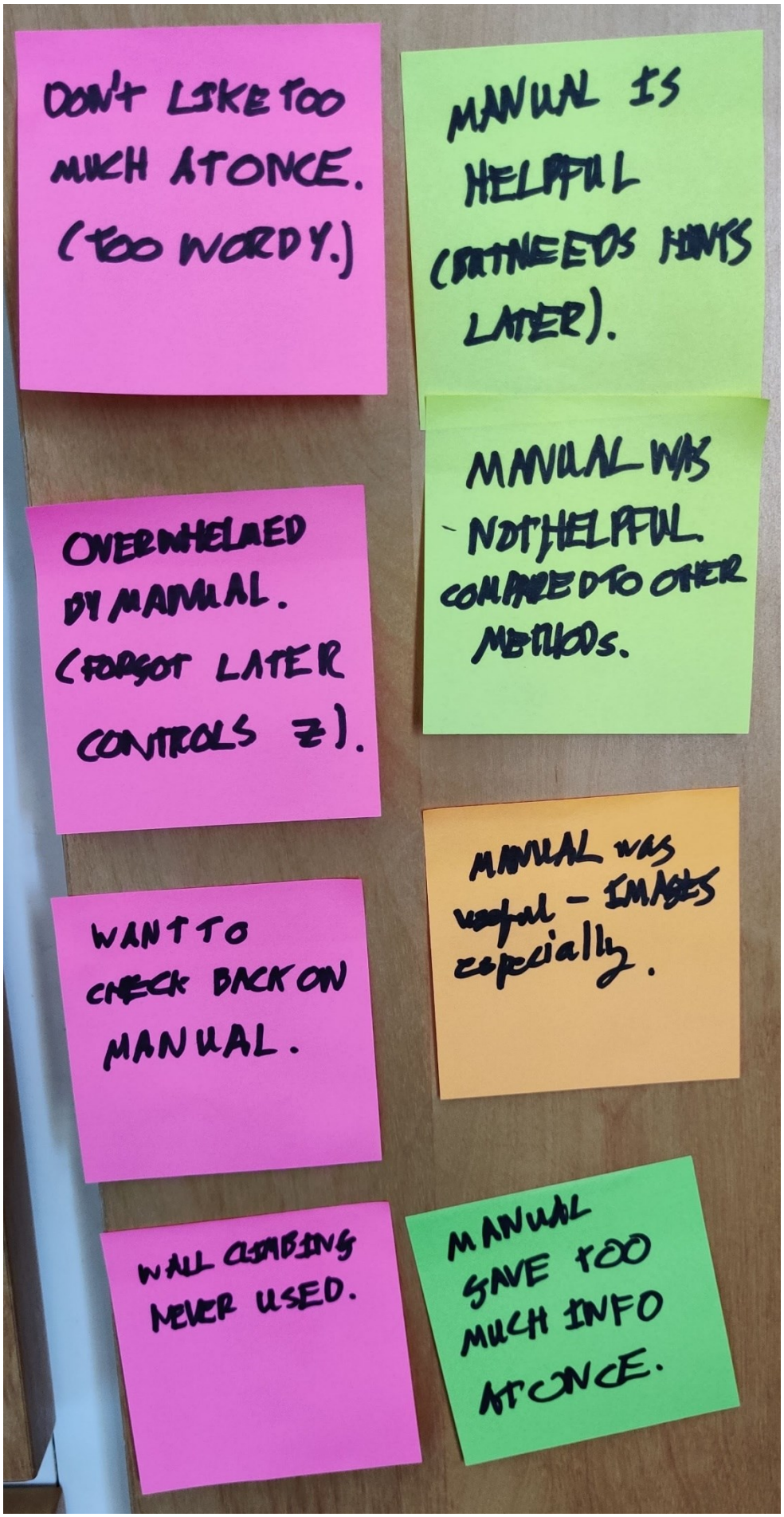
2 - Segmenting & Pre-Training


FELT LOST + FRUSTRATED. IN SHORT ET. DOESN'T FEEL ACHEIVEMENT
CONFIDENCE BUILT BY SEGMENTING (Longer Tutorial)
LONGER TUTORIAL BETTER. MORE ACHEIVEMENT.
CONFIDENCE GAINED BY COMPLETING LEVELS.
LONGER TUTorial. IMPROVED EXPLORATIVE LEARNING.
SHORT TUT. DROUGHT ON CONFUSION. LOSS OF FLOW.
SHORT LEVELS. TRIGGERED FEELINGS OF ACHEIVEMENT.
WANTS MORE CHECK-POINTS.
SEGMENTING IMPROVED EXPERIENCE.
NEED LESS PUNISHMENT IN LONGER LEVEL MORE RESPAWNS.
IMPORTANT TO GIVE TIME + SPACE TO IMPROVE.

3 – Level Design & Signifiers



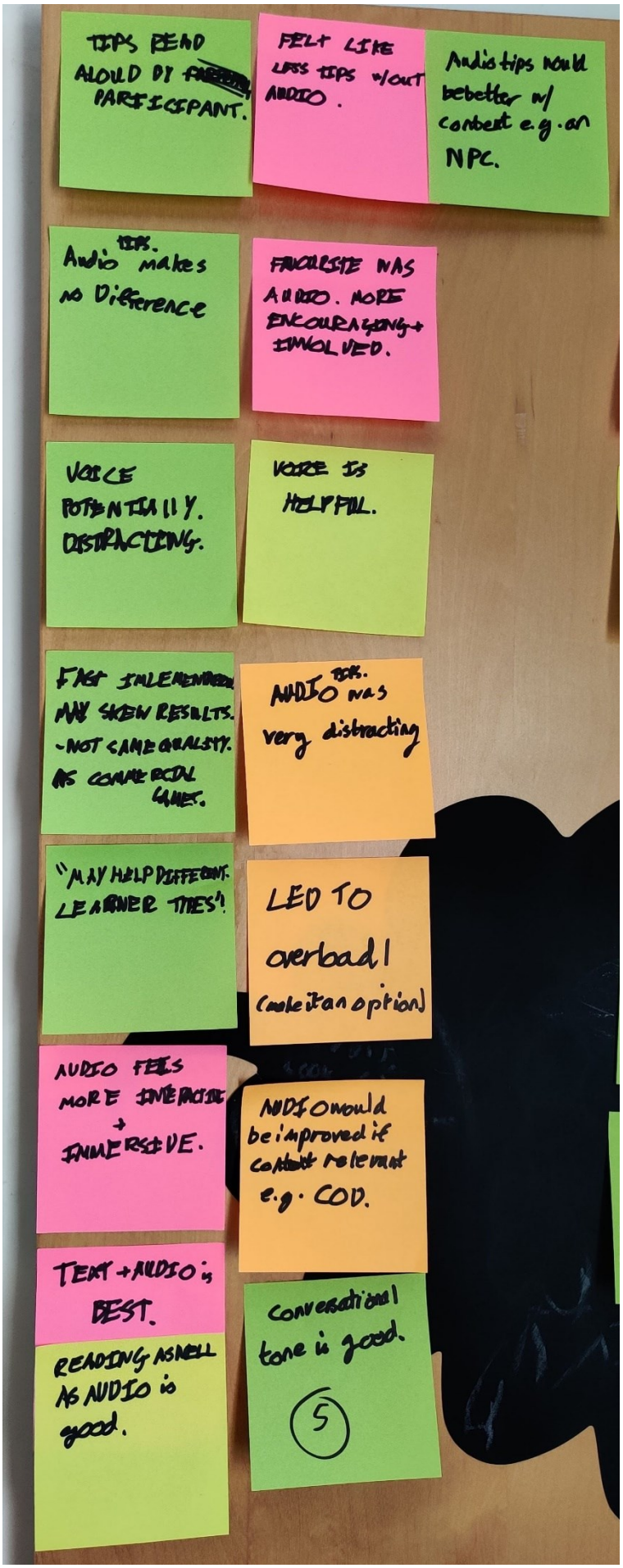
4 – Game Manuals



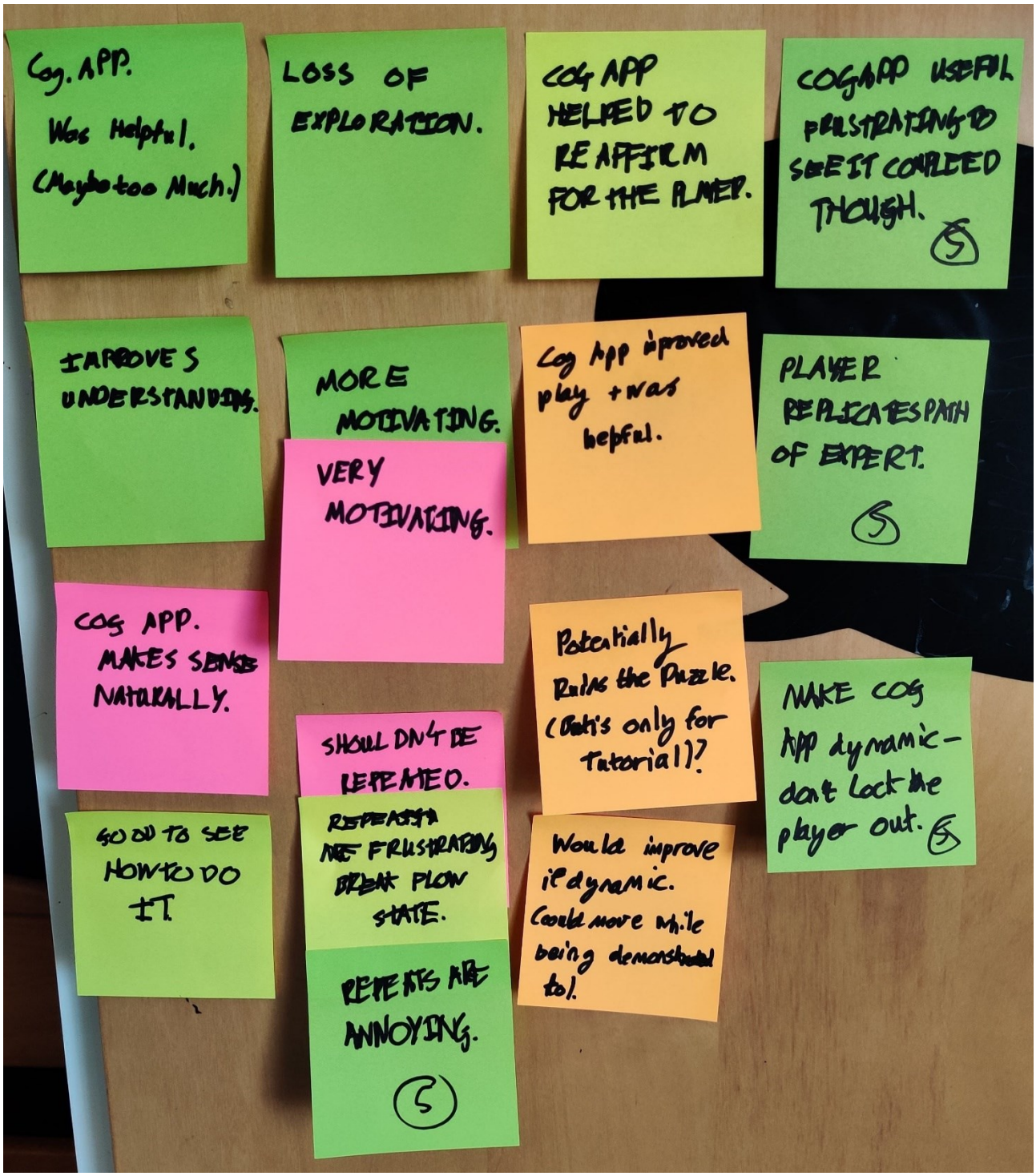
5 – Just In Time Instruction


JITI improved by (less time on each level).
NO SUCH THING AS TOO MANY CONTEXT RELEVANT HINTS.
TIPS ONE AT A TIME IS REALLY HELPFUL.
JITI was best – want to be able to check back on earlier Hints.
DEEPER CONVERSATIONAL TONE (More Vague Guidance).
ALWAYS MORE CONTEXT RELEVANT HINTS.
"GOOD TIMING IS ESSENTIAL" SCHEMA BUILT FROM PREVIOUS HINTS.
SCAFFOLDING WOULD BE HELPFUL.
JITI improve experience. more supportive
Pop ups more interactive + informative.
JITI text was best option.
JITI aided play + discovery. (5)
Want to check Back on Hints. (5)
Good that it dissapears when not needed (5)
GLAD JITI isn't immediate - gives player time to try before the hint. (5)
JITI is the best - (5) context relevant is important.

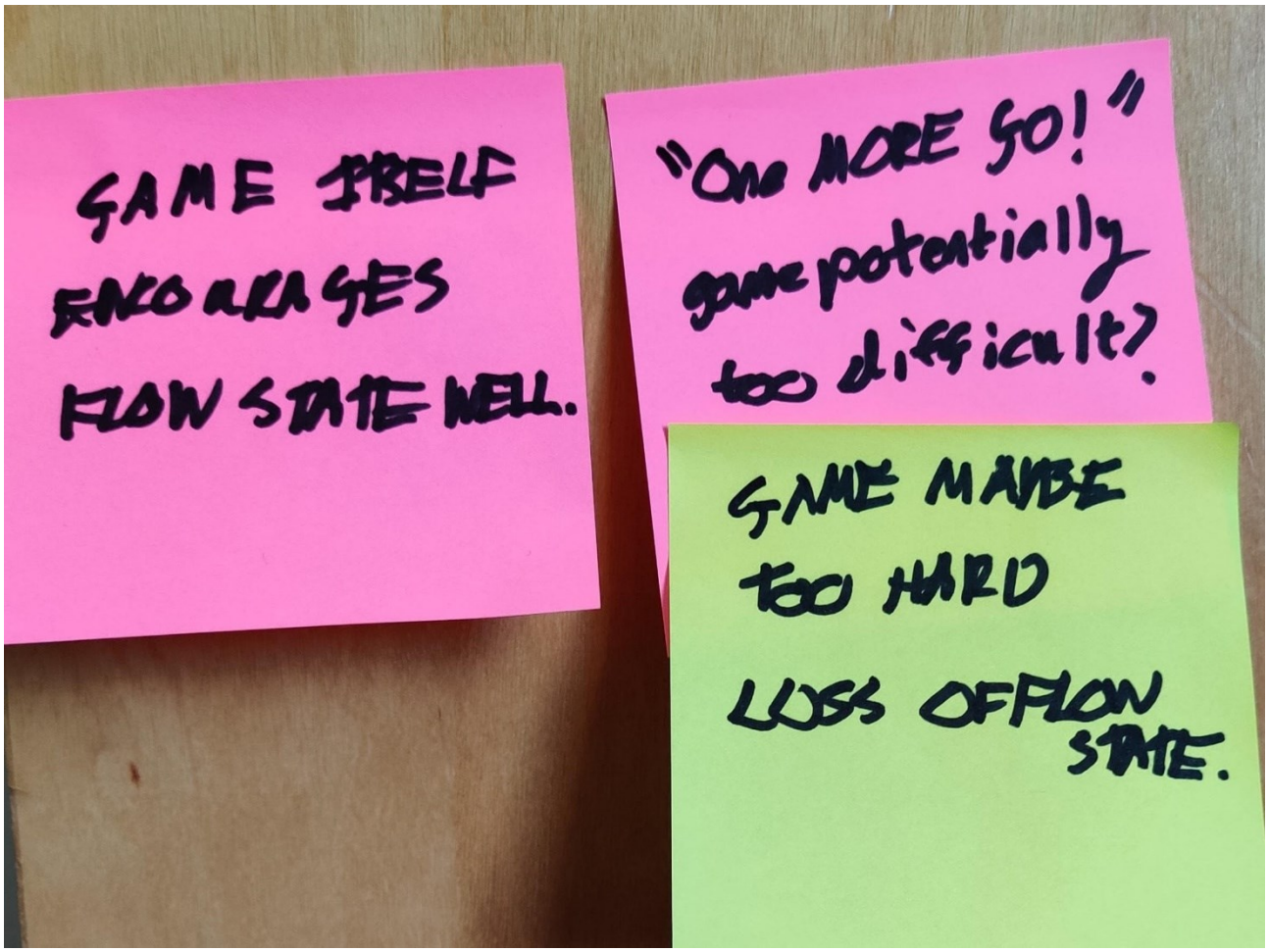
6 – Dual Coding & Audio Tips


TIPS READ ALOUD BY PARTICIPANT.
FELT LIKE LESS TIPS W/OUT AUDIO.
Audio tips would be better w/ context e.g. an NPC.
Audio tips makes no difference
FAVOURITE WAS AUDIO. MORE ENCOURAGING + INVOLVED.
VOICE POTENTIALLY. DISTRACTING.
VOICE IS HELPFUL.
FAST IMPLEMENTED MAY SKEW RESULTS. -NOT GAME QUALITY. AS COMMERCIAL GAMES.
AUDIO tips was very distracting
"MAY HELP DIFFERENT LEARNER TYPES"!
LED TO overload (make it an option)
AUDIO FELTS MORE INTERACTIVE + IMMERSIVE.
AUDIO would be improved if content relevant e.g. COD.
TEXT + AUDIO is BEST.
READING AS WELL AS AUDIO is good.
conversational tone is good.
5

7 – Cognitive Apprenticeship



8 – The Game Itself



9 – Additional Suggestions