3D Unity Game
One Last Look At The Sun
Interim Report

Contents

1 Introduction

1.1 Objectives

The objective of this project is to design, implement, and evaluate a 3D video game. The project will be made using the Unity 3D game engine [1] to run on the Windows 10 operating system.

In this game players must traverse down a seemingly bottomless pit in a first person perspective. The goal of this descent is to reach an end state upon which a playthrough of the game is successfully completed. Inside the pit obstacles exist to damage and hinder the player. The player is given tools to navigate these obstacles. Upon losing all health, the player's character will become incapacitated and the player must start again from the top of the pit. No in-game progress will persists through to the new attempt. To counteract this loss of progress, players will uncover cave paintings or "murals" which pictorially convey helpful information that will make subsequent attempts more successful. These murals may also convey story elements. The pit would be algorithmically generated anew on each attempt, meaning that instead of being able to memorise a preset sequence of obstacles, players must react to new situations in real time.

After implementation of the game has finished, user studies will be run to evaluate the experience of playing the game.

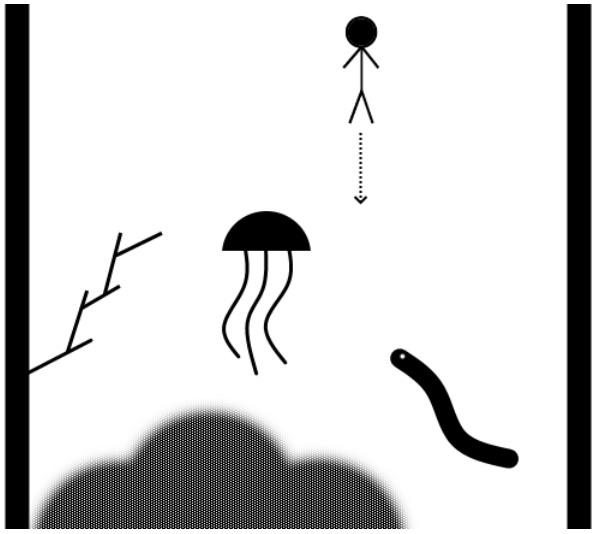


Figure 1: Representation of the player descending down a pit containing obstacles.

1.2 Motivations

The motivation of this project is that it will both give me experience at developing all aspects of a video game, and to have a portfolio piece to show to prospective employers.

This project will also allow me to put into practice material learnt from modules on my course. Project management techniques from Software Engineering will be heavily relied upon. Additionally, concepts learnt from Human Computer Interaction will be very useful in both the interaction design of the game and the user evaluation.

Finally Unity 3D uses C# for scripting which is very similar language to Java, a language in which the majority of my course is taught. The use of a similar language will allow me to be quickly proficient while also broadening my knowledge of languages.

1.3 Problem Area

The main challenges of making a game are the required time and skill set. There is always more work that could be done, and that work will be spread out across a wide variety of skills.

Even an outwardly simple seeming 2D game such as Downwell took 15 months of development by a single person [2]. The 3D action rhythm game Thumper was released 7 years after the start of development, even with two people working on it! [3]

1.3.1 Practical Problems

At the start of "The Art of Game Design" Schell lists a long list of skills needed for game development ranging from architecture to psychology[4, p. 2]. This list can be condensed to some core practical areas that we must succeed in:

- Visual arts - modelling, texturing, animation.
- Engineering - programming.
- Sound design - sound effects, music.
- Creative Writing - dialogue, characters, fictional histories.

I have limited experience in modelling, texturing or animation and these processes tend to take a long time regardless. The same can be said for both sound design and writing. As the time for the creation and evaluation of this project is very limited, careful consideration must be given to design decisions. So as to ensure there is enough time for completion.

1.3.2 Game Design Problems

In addition to concrete problems, there are more intangible game design problems that must be considered.

In "A theory of fun" Raph Koster talks about the idea of players, somewhat unknowingly, trying to optimise games. As a game is essentially a set of problems to solve, naturally players want to solve these problems as easily as possible. Or to put simply: find the optimal path. Unfortunately this optimal path is not necessarily (and is in fact rarely) the same as the most enjoyable path. Koster likens this to "An Alexandrine solution to a Gordian problem" in reference to the story where when Alexander the Great was faced with the unsolvable Gordian knot, he simply cut the knot [5, Chapter 7]. We can specify enjoyable and rewarding challenges to players, but that does not mean they will solve those challenges in the way we intend. Whether it be by a flaw of game design or an exploit in the game's code, players will try their hardest to cheat - both metaphorically and literally - themselves out of fun.

Another problem that leads out of this idea of optimising the difficulty out of challenges, is where does this difficulty even begin? If the game is no longer challenging, is it worth playing? Koster says "Games that are too hard kind of bore me, and games that are too easy also kind of bore me" [5, Chapter 1]. How are we going to set a difficulty that fits a large range of people?

2 Research And Related Works

2.1 Visual Arts

We noted above that acquiring visual assets was one of our key problem areas. Creation of assets by hand is time consuming and difficult. Some developers therefore choose to acquire assets from third parties.

2.1.1 Asset Flipping

A danger, however, of acquiring visual assets from elsewhere is falling into a practice called "Asset Flipping". The term was coined by games journalist Jim Sterling to describe the act of acquiring assets from third parties (sometimes without attribution or even the right to use), and then creating (and selling) games made solely from these assets with little to no original work. This does lead to incredibly fast development, but on top of the ethical dilemmas, games made in this way tend to be incoherent and of poor quality. Jim Sterling used The Slaughtering Grounds as a prime example. A game, which on top of it's sub par user experience, paradoxically features a London telephone booth situated next to a US post box. [6].

Figure 2: The Slaughtering Grounds, a notoriousl "asset flip".

2.1.2 A Focus On Visual Effects

A much better solution to the problem of acquiring visual assets could be to pursue an art direction where we create relatively simple models (in Blender[7], a 3D modelling program) and use programmatic visual effects to embellish these models. By transforming this problem into a task of engineering, and therefore playing to my strengths, it will be possible to deliver a compelling aesthetic.

There are visual effects programs called shaders that be can written to run on graphics hardware accelerators. A fragment shader could be used to generate and apply a texture to a model [8, p. 129], or a vertex shader could be written to move the vertices of a 3D model in such a way that it animates the model[8, p. 113]. While these textures and animations would not be suitable for all objects, you could not animate and texture a human solely with mathematical functions, entities could be designed with this constraint in mind. You could, for example, animate a swimming eel by displacing vertices based on the sine function of each vertex's world position.

This is exactly what the developers behind Thumper did. Nearly all of the game's animations and textures are generated by shaders [3]. Brian Gibson, one of the two developers of Thumper, stated "The visual look is in many ways the result of custom vertex shaders we built to create perfect bends on the path and tentacles in the world" [9].

Other visual effects like particle systems would also be very useful. Particle systems can both function as atmospheric effects (e.g. rain and fog) and as entities. Imagine a system of glowing particles acting as fireflies.

2.2 Engineering

Another key problem area mentioned was engineering. Here we want to reduce the amount of work needed to implement the functionality of the game, whilst

Figure 3: Thumper, a game that manages to deliver a compelling aesthetic despite having simplistic models and a lack of any textures due to its use of visual effects and shaders.

not significantly reducing it's quality.

2.2.1 Engine

The most obvious respect in which time can be saved is using a pre-built game engine rather than coding one from scratch. I chose to use the Unity 3D engine for two reasons:

- Use of the C# language for scripting, a language very similar to Java, in which the majority of my course was taught.
- The engine is very popular, meaning it has a rich feature set and a wealth of support material online.

2.2.2 State Persistence

Another time consuming task is implementing the systems needed to store a game's state persistently across play sessions. It, therefore, might be beneficial to have a design in which these elements would not be required. A game that does this is Outer Wilds, in which you have 22 minutes to save a solar system before it gets destroyed. After those 22 minutes are up (or upon death), the world is completely reset. Nothing persists through apart from the player's knowledge about the world and that knowledge is the only progression in the game. It is more than possible to complete the game in those 22 minutes, but only if you have acquired the knowledge to do so.

2.2.3 Interfaces

With a lack of persistence mechanics, the need for menus and UI elements is reduced. This lack of need could be further exploited so that these interface elements would not have to be implemented at all. CELL, a music game made by a previous third year Sussex student, does not use any menus or UI elements, all information in the game is elegantly communicated from within the game world [10].

2.2.4 Procedural Generation

One way to cut down on the effort needed to make levels or scenarios for the game would be to procedurally generate them. We feed a random seed into a system and the system outputs, based on a series of written rules, a level for the player to complete. This provides nearly endless content.

A problem, however, with procedural level generation is that the levels produced tends to be of a much lower quality than authored content.

The game Spelunky overcame this by implementing a compromise system. Bare bones templates of level sections are hand made, each of these templates with a specific challenge for the player, and an intended way of overcoming that challenge. Templates are then combined procedurally to construct a whole level. An algorithm then adds variation to these templates by populating them with different game objects [11, p. 37]. This means that there is a huge space of possible levels, and less chance of players getting bored. Yet these levels don't lose the feeling that they were designed with specific interactions in mind.

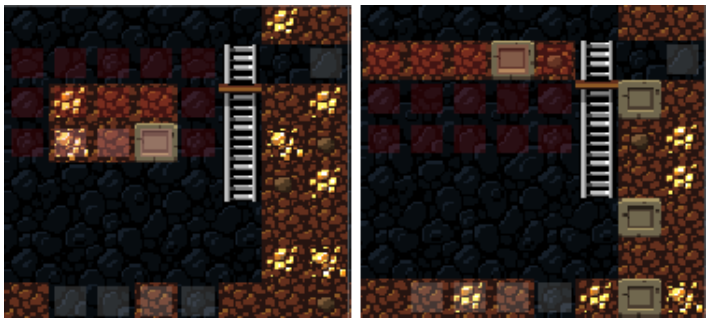


Figure 4: Two variations on the same level template in Spelunky.

2.3 Sound Design

Creating sounds for a game can be a long and trying process. Large games have entire teams devoted to crafting a soundtrack. This is without even considering all the sound effects needed to give interactions a feeling of impact.

A strategy to lessen the workload would be to partially combine music and sound effects. The sounds played upon interactions in the game could be musi-

cally pleasing to the ear. Multiple effects played in succession could form a sort of a melody. Then this "melody" could be supported by minimalist background tracks. A form of this is employed by the previously mentioned game Thumper. Upon successively neutralising obstacles an acoustically pleasing, rising melody plays. This also doubles as effective feedback, it is immediately obvious when a mistake is made as the melody suddenly becomes discordant.

A method to generate these sound effects could be by recording small audio clips via a microphone and manipulating them in the audio editing software Audacity[12]. Effects like "paulstretch" take innocuous sounding samples and add layers of haunting ambience at the press of a single button. Other effects allow pitch and tone shifting to make samples sound more musical.

An effective way to generate the background tracks would be to define rhythms in TidalCycles[13], a Haskell based sample engine, then feed these rhythms into VCVRack[14], a modular synthesis emulator, to add detail and texture.

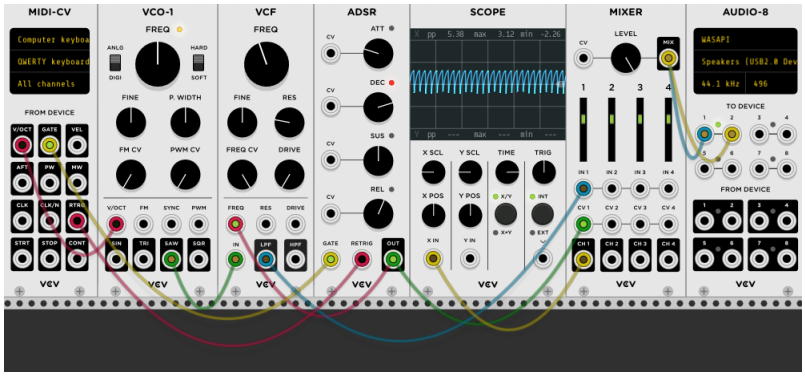


Figure 5: VCVRack, modular synthesis software.

2.4 Creative Writing

Writing good in-game dialogue is a lengthy process and presents issues with localisation to different regions. To counter this, an attempt could be made to communicate as much information as possible visually. Any story elements could be more abstract, prioritising atmosphere above concrete narratives. In the game Journey, all story elements are communicated to the player solely via gameplay and visual cut-scenes. Yet according to the creative director of the game, Jenova Chan, the game managed to make 3 of the 25 final version testers cry upon completion[15].

2.5 Game Design

The two problem areas we identified in relation to game design were the ideas of: the optimal strategy of a game not necessarily being the fun strategy, and

Figure 6: Journey, a game that manages to effectively convey emotion effectively without the use of language.

how to decide how difficult to make the game.

2.5.1 The Optimal Strategy

In the game Bloodborne, you are a warrior tasked with defeating terrific beasts. When unexpectedly hit by a foe, most players tend to lose their composure. This means that they are worse at decision making and reacting to situations. As the foe has spotted an opening, he's likely to attack again immediately.

The optimal strategy here might appear to be to run away, regain your composure and then re-engage the enemy. And this is completely the case in previous similar games made by the same development studio. This optimal strategy is passive, it ruins the tempo of the fight, and removes any sense of urgency. This optimal strategy is not a fun strategy.

However in Bloodborne, after taking damage from an enemy, there is a short period of time where any successful attacks restore some of your lost health. This changes the optimal strategy to being that you should immediately retaliate. The game has just watched you suffer the consequence of taking a risk, but instead of telling you run and hide, it convinces you to take yet another risk - and taking risks is fun.

2.5.2 Difficulty

A simple answer would be to how difficult to make a game is to simply let players choose from a set of difficulty presets at the beginning of the game. This a widely used solution in industry.

Unfortunately it is very difficult to scale a game's difficulty in a meaningful way. A lot of games will simply just reduce your health and increase the enemies health, or increase the number of enemies you must defeat to progress. While

this sound suitable at first, these changes can actually change the way players may interact with the game. They may make the "fun" strategy ineffective. Some games are even aware of this, and will recommend the game be played at a specific difficulty. Halo 3, a popular shooter game, has four difficulties but the description for the "Heroic" difficulty reads "...this is the way Halo is meant to be played".

The game Downwell has no difficulty settings and instead uses a concept known as player-modulated difficulty. It is the idea that a player will make the game easier or harder for themselves in real time by the choices they make during play. As you are falling down the titular well in the game, you must defeat enemies. If you are struggling with the game you may choose to descend slowly with care. However, if you have a higher ability this slow pace might bore you. So the game offers you two challenges: how fast can you descend down the well and how many enemies can you defeat without touching solid ground (a combo). Succeeding at these challenges rewards you with items, that you don't *need* to complete the game, but are nonetheless useful. This turns easy levels from a bore to an opportunity to practise difficult mechanics.

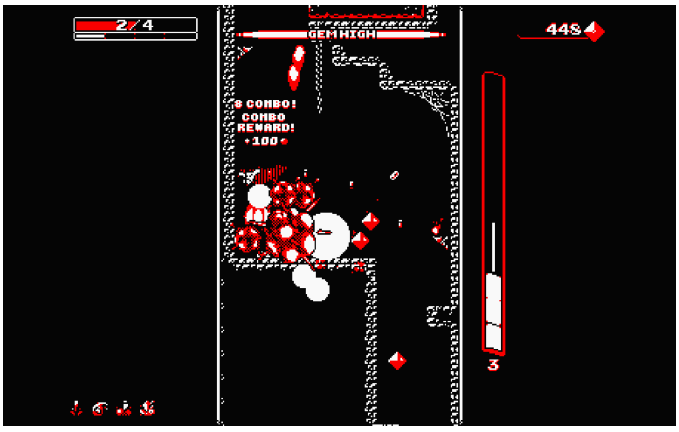


Figure 7: Downwell, a player being rewarded for both completing a combo and for being "gem high" (descending at a faster rate).

3 Evaluation

During development expert feedback would be obtained via heuristic evaluation of the game against a set of external criteria [16, p. 550]. This would be an iterative process, where feedback is gathered, changes are implemented, and then these changes are shown back to the expert for more feedback. The number of iterations would be entirely time dependant.

Then once the implementation of the game is finished, a user experience test will be run. First, there would be a direct observation in the field [16, p. 288]

of participants playing the game. It is important to see if players interact with the game in the way intended without any knowledge external to the game.

After playing the game, a semi structured interview [16, p. 269] would be run with participants to ask about their experiences in both qualitative ("How was your experience of the game?") and quantitative ("On a scale of 1-10 how fun would you rate the game?") formats. Participants would also be encouraged to voice any questions or suggestions they might have in the interview debriefing script.

4 Professional and Ethical Considerations

4.1 BCS Code of Conduct

The relevent sections of the BCS Code of Conduct are as follows:

1.1 - have due regard for public health, privacy, security and wellbeing of others and the environment;

The game does not feature distressing imagery, the only data held about the player may be the speed at which they completed the game, if consented to by the player.

1.2 - have due regard for the legitimate rights of third parties;

The game makes very limited use of third party content and where it is used, the content is free to use, all license obligations have been met, and the creator is properly attributed .

1.3 - conduct your professional activities without discrimination on the grounds of sex, sexual orientation, marital status, nationality, colour, race, ethnic origin, religion, age or disability, or of any other condition or requirement;

Characters within the game do not outwardly possess the aforementioned traits nor is the identity of the user relevant to the playing of the game. Therefore the game cannot promote or exercise discrimination based the these features.

1.4 - promote equal access to the benefits of IT and seek to promote the inclusion of all sectors in society wherever opportunities arise.

Efforts will be made to accommodate users with specific disabilities (E.g. colour blindness, deafness, motor disabilities).

2.4 - ensure that you have the knowledge and understanding of legislation and that you comply with such legislation, in carrying out your professional responsibilities;

Once the project's implementation has been finished, the Pan European Game Information (PEGI) guidelines will be consulted to determine the age suitability of the game.

2.5 - respect and value alternative viewpoints and seek, accept and offer honest criticisms of work;

Results of all expert feedback and user testing shall be presented without edit or omission.

2.7 - reject and will not make any offer of bribery or unethical inducement.

All feedback shall be gathered without any incentive based on the sentiment of the feedback.

4.2 Ethical Review

As this project requires feedback from human participants consideration of ethics is required.

1. Participants were not exposed to any risks greater than those encountered in their normal working life.

The project is a non-violent video game and does present any significant risks.

2. The study materials were paper-based, or comprised software running on standard hardware.

Participants will both play the game and fill out any feedback forms on my personal laptop (i.e. on standard hardware).

3. All participants explicitly stated that they agreed to take part, and that their data could be used in the project.

Participants will be briefed on the project, what is required of them and what information will be collected and stored before the user test begins.

4. No incentives were offered to the participants.

Participants will be volunteers and not receive any incentive to participate or to give feedback of a specific sentiment.

5. No information about the evaluation or materials was intentionally withheld from the participants.

The nature of the project being a video game means that there will be information that the user must discover while playing the game. It is important to observe how users interact with the game with little prior knowledge, so in this sense some information will be withheld. However, participants will be aware that they are being observed, be briefed on the project goals, and be offered an opportunity to ask questions before the user test begins. There will be no deception of any kind.

6. No participant was under the age of 18.

Participants will be students of the university and thus all of age 18 or above.

7. No participant had a disability or impairment that may have limited their understanding or communication or capacity to consent.

Participants will be students of the university (i.e. having the cognitive capacity for university-level courses) and user tests will conducted on campus grounds during the working day (i.e. not sleep deprived, or otherwise unable to properly consent).

8. Neither I nor my supervisor are in a position of authority or influence over any of the participants.

Participants will be my fellow students and while my supervisor is in a position of authority as part of the faculty, the identities of participants will be not known to anyone but myself.

9. All participants were informed that they could withdraw at any time.

The ability to withdraw at any time will be stated in the brief.

10. All participants have been informed of my contact details, and the contact details of my supervisor.

Paper slips containing the email addresses of both myself and my supervisor will be offered to participants during the briefing and debriefing.

11. The evaluation was described in detail with all of the participants at the beginning of the session, and participants were fully debriefed at the end of the session. All participants were given the opportunity to ask questions at both the beginning and end of the session.

A briefing and debriefing script containing such will be used to conduct all user tests.

12. All the data collected from the participants is stored securely, and in an anonymous form.

All response forms and observation notes will be stored securely on my personal password protected laptop with any identifiable information redacted.

The project meets the 12 criteria listed in the Ethical Compliance form and a signed copy of this form is included as an appendix.

5 Requirements analysis

The goal of this project is to deliver an enjoyable user experience, targeted at any user who might be interested in such an experience.

5.1 Usability

In Julian Gold's book "Object Oriented Game Development" he states some common priorities of games, two of which are: speed and robustness [17, p. 16].

Speed being that each subsequent frame is drawn to the screen soon enough after that last that we perceive it as a fluid motion, rather than a series of images. We will need to make sure the game runs acceptably on common hardware. For ease, we shall define common hardware here as my development laptop (ASUS GA502DU).

Robustness meaning that the game doesn't crash or feature game breaking bugs causing the user to unfairly fail. While we can't remove all bugs, we shall set an acceptable threshold.

We must also define a target system for the game to run on and input methods.

5.2 Gameplay

The game must be fun. The Art of Game Design defines fun as "pleasure with surprises" [4, p. 26]. To continually surprise the player, we must have a suitable amount of distinct content. As we are using a hybrid of hand made content and procedural content, we need an adequate amount of templates for our algorithm to work with.

There are also concrete requirements specific to the game proposal and more vague concepts mentioned in the related work. One idea not mentioned previously is to include features that would encourage a practise called speedrunning, where players compete to finish a game in the fastest time possible. This would give players a reason to play again even after completion.

5.3 Accessibility

An often overlooked area is the need for accessibility. In fact none of the literature cited so far even includes the word "disability". A 2011 paper estimated that as much as 9% of the (US) population had disabilities that would limit the enjoyment or games and as much as 2% had disabilities that would make most games unplayable [18]. Effort should be made, if possible, to accommodate such a significant number of people.

5.4 Requirements

5.4.1 Mandatory

1. The game runs on the Windows 10 operating system

2. The game runs at an average frame rate of 30 frames per second on common hardware.
3. The game experiences a crash or bug which causes the player to fail at a rate of less than once an hour.
4. The game shall accept user input by both a keyboard/mouse and an XInput gamepad.
5. The player shall descend through a pit.
6. The player shall be able to move laterally.
7. The game shall be played from a first person perspective.
8. The player shall have access to at least one tool to neutralise specific obstacles.
9. The player shall encounter obstacles that damage or hinder the player.
10. There shall be at least 10 unique obstacles.
11. The pit should be procedurally generated.
12. There shall be at least 25 pit section templates.
13. The player shall encounter more challenging content as they descend further into the pit.
14. The player shall encounter pictorial murals that convey helpful game information.
15. There shall be a maximum depth that upon reaching signals that the player has successfully completed a playthrough of the game.
16. There shall be sound effects for all player interactions in the game.
17. There shall be background music in the game.
18. There shall be a visual intro sequence to give context about the game.

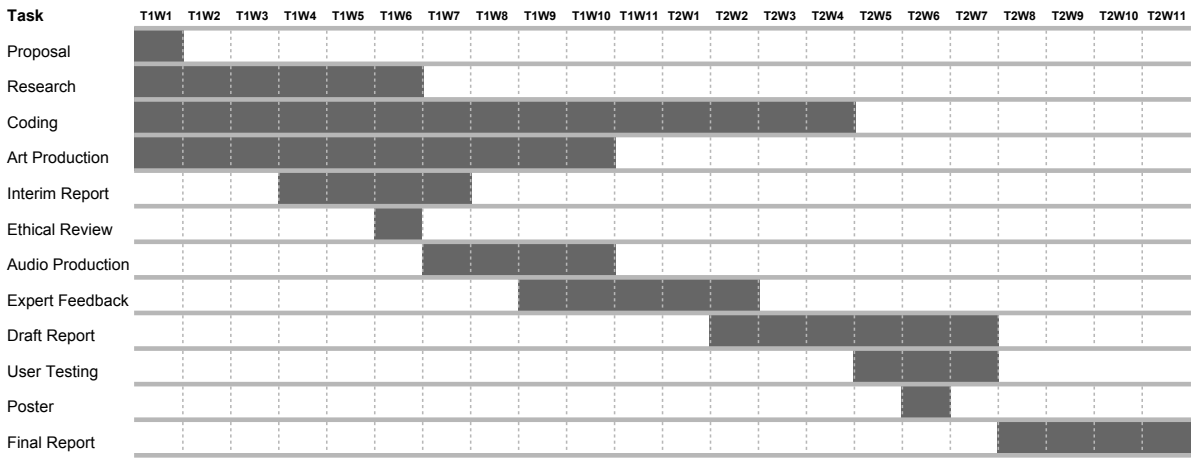
5.4.2 Desirable

19. There should be a mobile version of the game.
20. There should be Mac OSX and Linux versions of the game.
21. There should be at least 15 unique obstacle types
22. There should be at least 50 unique pit section templates.
23. Players should be offered incentives to play in more challenging and risky ways.

24. Players should be able to uncover secrets in murals that let them skip early, less challenging part of the game.
25. There should be story elements for the player to witness and uncover.
26. There should be personal milestones for completing the game in under defined times.
27. There should be an online database that (optionally) stores the times of the fastest completions of the game.
28. There should be **unique** sound effects for most entities and interactions in the game.
29. The game should make efforts to be accessible to persons with auditory disabilities (playable with no sound).
30. The game should make efforts to be accessible to persons with visual disabilities(playable with various forms of colour blindness).
31. The game should make efforts to be accessible to persons with motor disabilities(remappable, simple controls).

I expect to meet all of the mandatory requirements and to partially complete several of the desirables.

6 Project Plan


Task
T1W1
T1W2
T1W3
T1W4
T1W5
T1W6
T1W7
T1W8
T1W9
T1W10
T1W11
T2W1
T2W2
T2W3
T2W4
T2W5
T2W6
T2W7
T2W8
T2W9
T2W10
T2W11
Proposal
Research
Coding
Art Production
Interim Report
Ethical Review
Audio Production
Expert Feedback
Draft Report
User Testing
Poster
Final Report

6.1 Tasks

- Proposal - Description of proposed project submitted to supervisor, included in appendices.
- Research - Survey of games, conference talks, books and scholarly articles relevant to this project.
- Coding - All programming/engineering tasks, e.g. player movement, endless pit generation, obstacle logic, procedural object placement.
- Art Production - All visual tasks, e.g. modelling, creating shaders for texturing/animation, title screen, visual intro sequence.
- Interim Report - See above.
- Ethical Review - Checking that the project meets the ethical compliance guidelines with supervisor and writing justifications for each of the points. Signed form included in appendices.
- Audio Production - Background music and interaction sound effects.
- Expert Feedback - Iterative process of heuristic evaluation by expert.
- Draft Report - First version of final report, documenting whole project.
- User Testing - Running in-person user experience tests with voluntary participants.
- Poster - Design of poster for poster event.
- Final Report - Implementing suggested changes from supervisor feedback of draft report.

In the implementation phase the tasks of coding, art and audio production shall be run in parallel. Each is made up of many smaller discrete jobs and it may be beneficial to perform jobs from different tasks that relating to the same idea (e.g. a specific obstacle) sequentially.

Additionally this parallelism benefits the process of expert feedback. The expert will be able to evaluate code, art, and audio together with the schedule also allowing time for subsequent changes to all three.

Coding is scheduled to extend past art and audio production. The last stages of implementation should be focused on polish and bug fixing. It is more important that the game functions well rather than having a few extra features.

By the time user evaluation starts, work on the game must stop. All users should experience the same version of the project. During this stage, the implementation sections of the draft report can be written.

Once the user evaluations have been completed they can be added to the draft report, and the draft can be send for feedback.

Once feedback is received the final report can be written.

6.2 Completed Work

Research of related works and literature has been mostly completed, small amounts of further research may be required further into the implementation stage and when writing up evaluation materials.

The proposal was completed very early on and is attached in the appendices.

A base set of functionality has been implemented: the player falls through a seamless unending pit, with a simplistic form of procedural obstacle placement. The player has two tools to navigate these obstacles: a dash and a block ability, however these are very much subject to change. Partially completed obstacles include:

- Jellyfish - floats upwards, damages the player if touched.
- Eel - chases the player, must be blocked.
- Toxic cloud - stationary cloud, damages the player if touched.
- Luminescent plankton - jittering particles, must be blocked.
- Static lightning - periodically damages the nearby area.
- Skate fish - swims in a circle, damages the player if touched.
- Wall tentacle - swings from side to side, damages the player if touched.

A simple title screen and visual intro sequence have been implemented.

Finally some user feedback systems have been implemented, such as showing when the player has been damaged or when they are using the block ability.

Figure 8: Project title screen.

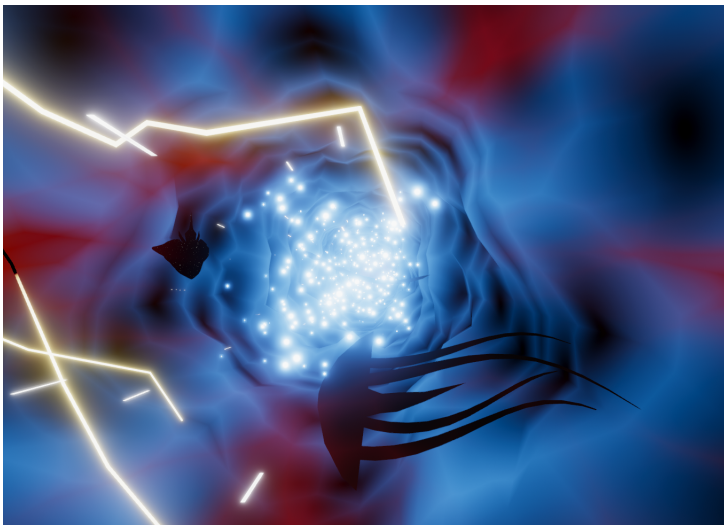


Figure 9: Project progress: lightning, skate, plankton, jellyfish, hurt status effect.

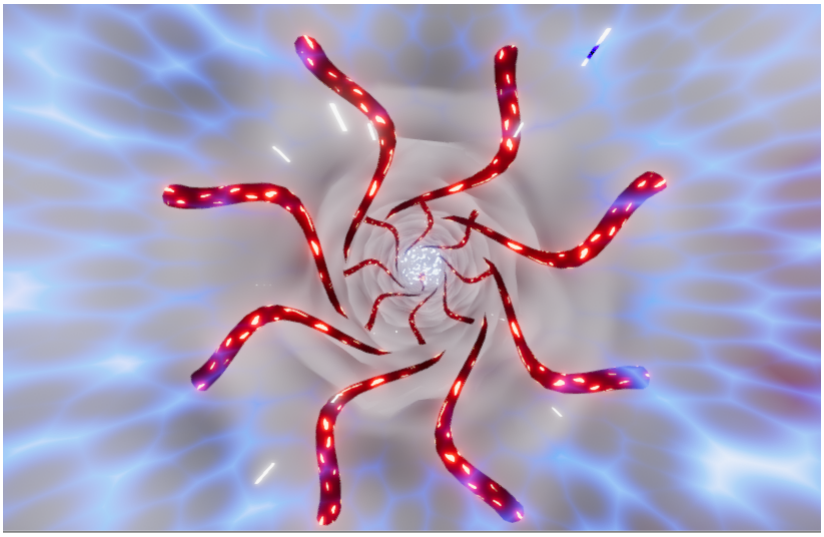


Figure 10: Project progress: tentacles, shield/block ability.

6.3 Other Tools

I have also utilised Trello for planning and interacting with my supervisor, and a private Github repository for version control.

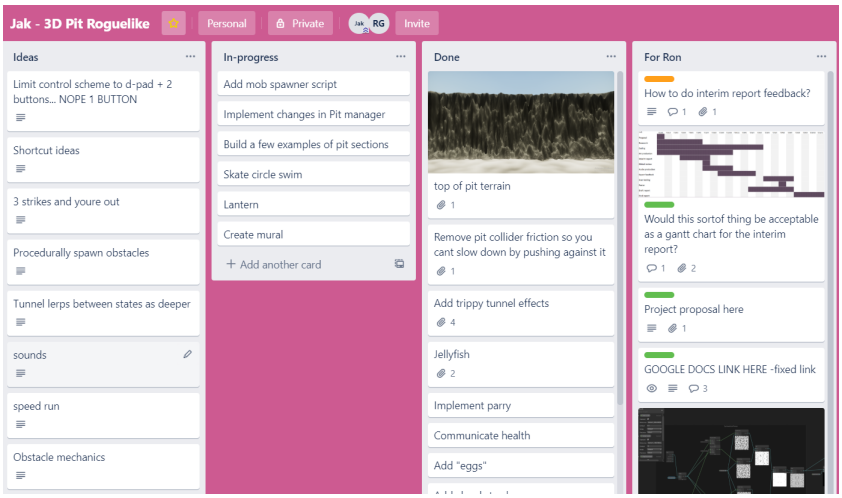


Figure 11: Trello project page.

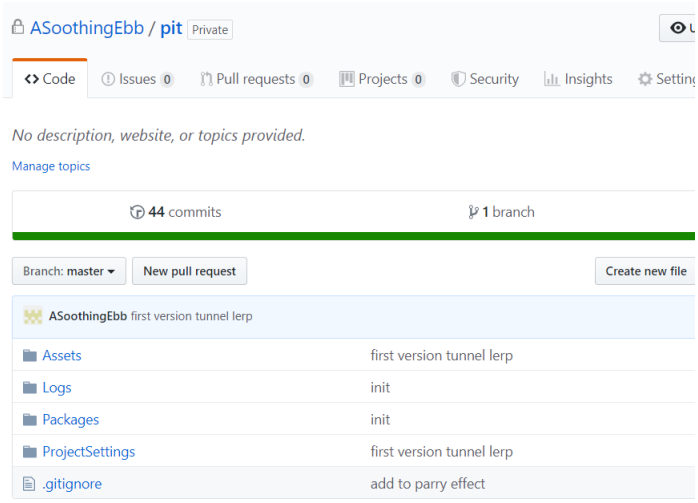


Figure 12: Github project repository.

7 Referenced Games

Title, Developer, Publisher, Year

Outer Wilds, Mobius Digital, Annapurna Interactive, 2019

The Slaughtering Grounds, Imminent Uprising, Digital Homicide Studios, 2014

Thumper, Drool, Drool, 2016

Journey, Thatgamecompany, Sony Computer Entertainment, 2012

Bloodborne, FromSoftware, Sony Computer Entertainment, 2015

Halo 3, Bungie, Microsoft Game Studios, 2007

Downwell, Moppin, Devolver Digital, 2015

References

[1] Unity Technologies. Unity 3d. https://unity.com/.

[2] Ojiro Fumoto. Polishing the boots - designing 'downwell' around one key mechanic. Game Developers Conference, 2016.

[3] Marc Flury. Seven years in alpha: 'thumper' postmortem. Game Developers Conference, 2017.

[4] Jesse Schell. *The Art of Game Design: A book of lenses.* CRC Press, August 2008.

[5] Raph Koster. *Theory of Fun for Game Design.* "O'Reilly Media, Inc.", 2005.

[6] Jim Sterling. The asset flip (the jimquisition). `https://www.youtube.com/watch?v=5svAoQ7D38k`, accessed 05/11/2019.

[7] Blender. https://blender.org/.

[8] Mike Bailey and Steve Cunningham. *Graphics shaders: theory and practice.* CRC Press, 2012.

[9] Jennifer Allen. Gamasutra. `https://www.gamasutra.com/view/news/282867/Why_Thumpers_twist_on_rhythm_games_is_so_wellsuited_for_VR.php`, accessed 02/11/2019.

[10] Rob Dawson. Cell: A generative music game, 2013.

[11] Derek Yu. *Spelunky: Boss Fight Books #11.* Boss Fight Books, March 2016.

[12] Audacity. https://www.audacityteam.org/.

[13] Tidalcylces. https://tidalcycles.org/.

[14] Vcvrack. https://vcvrack.com/.

[15] Jenova Chan. Theories behind journey. D.I.C.E Summit, 2013.

[16] Helen Sharp, Yvonne Rogers, and Jenny Preece. *Interaction design: beyond human-computer interaction.* Wiley, 2019.

[17] Julian Gold. *Object-oriented Game Development*. Pearson Education, 2004.

[18] Bei Yuan, Eelke Folmer, and Frederick C Harris. Game accessibility: a survey, 2011.

8 Appendices

Proposal

Candidate number:

Supervisor:

Working title: 3D Unity game - One Last Look At The Sun

Aims and objectives

The project would be to create a 3D game where the player must traverse/fall down a seemingly bottomless pit, dodging/neutralising obstacles on the way, to uncover the secrets hidden within the pit. On a failed attempt the player must start from the top of the pit, however clues scattered inside from previous inhabitants of the pit (as well as knowledge gained simply from playing) would make subsequent attempts easier and quicker. In this way the game would become less challenging the more you explore (i.e. the difficulty is player modulated) and players do not have to spend as much time in sections of the pit they have already completed. Elements of the pit would be procedurally generated, changing every attempt, so that a player could not simply memorise where dangers are and is instead forced to adapt to problems in real time. As the player descends further into the pit the difficulty would increase in two main ways: firstly, a greater variety of more complex obstacles (e.g. falling debris, creatures, toxic fog) and secondly, perceptual challenges (e.g. visual distortions, auditory hallucinations, changes in the perception of time).

All assets would be sourced royalty free or be self-made.

The game would be built using the Unity game engine, the primary targeted platform being Windows with Linux and Mac OSX versions being desirable. The software would be delivered as a compressed folder containing an executable that could be immediately run.

Evaluation of the game would be via in-person and online user testing. In-person testing would allow observation of how users interact with different elements of the game and whether the game encourages them to play in the intended way. Online testing would let users spend as much time as they wanted with the game and might reach a greater audience. Testing responses would be gathered via a short survey where users can rate different aspects of the game and optionally leave comments.

Primary objectives
The game shall run on Windows 10.

The game shall be 3D.
The game shall consist of a series of procedurally ordered “pit sections”, meaning players are very unlikely to experience identical playthroughs.
The game shall include findable cave paintings or murals that will pictorially demonstrate complex or hidden game mechanics.
There shall be at least 20 unique pit sections each containing a series of obstacles (and murals).
There shall be at least 7 unique obstacle types.
There shall be a maximum depth that, once reached, means the player has successfully completed a run of the game.
The player shall be more likely to encounter difficult pit sections as they get deeper.
The player shall be able to interact with the game via both keyboard and Xinput gamepad.

Extension objectives
There should be a mobile version of the game.
There should be an online database to record fastest game completion times.
The game should be accessible to persons with auditory (playable with no sound), visual (playable with various forms of colour blindness), and motor (remappable, simple controls) disabilities.
The game should be stable while running (crashes or game breaking bugs at maximum rate of 1 per 2 hours of play)
There should be story elements, including an area to explore at the bottom of the pit.

Relevance

The unity game engine uses C# for scriptings, C# is very similar to Java which has been taught and assessed in many modules during my course.

Techniques learnt from the module Software Engineering can be used to manage the project and make sure a satisfactory product is delivered.

The project will allow me to become better at game development and have a portfolio piece, both would be useful for applying to game development jobs in the future.

Resources required

- Personal Laptop
- Unity 3D game engine Personal edition (Free)
- Blender modelling software (Free)

Timetable

	Monday	Tuesday	Wednesday	Thursday	Friday
9:00			ICS Lecture	ICS Lecture	
10:00	Project Work	Project Work	HCI Seminar	ICS Lab	CoP Lab
11:00	Project Work	Project Work	HCI Group	ICS Lab	Project Work
12:00	CoP Work	ICS Work	HCI Group	HCI Work	Project Work
13:00	CoP Work	ICS Work	Project Work	HCI Work	Project Lecture
14:00	Any Work	Any Work	Any Work	Any Work	Any Work
15:00	Any Work	Any Work	Any Work	Any Work	Any Work
16:00	HCI Lecture				CoP Lecture
17:00					CoP Lecture

Related student projects

Anonymised

Project Log

Preterm

My previous use of unity had been very limited so I experimented with the engine so that I wouldn't be completely lost once the project started. I specifically spent a lot of time trying:

- Shader graph, a visual scripting tool for shaders, instant feedback on changes, very quick to prototype ideas, especially useful as a way to combat my weakness at texturing/drawn art/animation.
- Visual effect graph, a visual tool for GPU based particle effects, same strengths as Shader graph.
- Creating, manipulating and destroying objects via C# scripting.

I watched most of the Game Maker's Toolkit videos on game design, this provided a lot of ideas about game mechanics I could use as well as frameworks to use to evaluate my own ideas. I also watched a few GDC talks such as the talks on Thumper and Downwell which inspired some of the main ideas for my project.

I learnt the basics of modelling in Blender: basic transformations, extruding, mirroring, UVs.

I played around with using Audacity to manipulate samples captured via microphone into possible sound effects for the game. I experimented using TidalCycles to generate rhythms that I then distorted in VCVRack which resulted in some simple music tracks.

For the rest of the time I researched games and read game design/implementation books until I had a vague outline of the game I wanted to create.

End of Term 1 Week 1

- Create a first draft of a gantt chart outlining creation of the game, submittable pieces of written work (e.g. ethics application), user testing, report writing etc.
- Wrote the first draft proposal to consolidate ideas for my project, from this I could begin experimenting with the feasibility of some ideas
- Made a simple representation of the player, a capsule rigid body that experiences gravity and has a camera attached
- Created a draft input map (movement, dash, parry) and attached the movement controls to applying forces on the player rigidbody, making a rudimentary movement system.
- Created a pit "section", a hollow cylinder with no end caps

- Wrote a C# to queue pit sections below each other as the player fell through them, giving the effect of a never ending pit (although you could see the bottom.
- Created a simple shader to colour and distort the mesh of the tunnel section based on a 2D noise function offset by time to simulate the look of clouds. Unfortunately this lead to a seam on the mesh where the two sides of the noise texture met (think of rolling up a textured piece of paper where the ends aren't exactly the same) and a seam where two tunnel sections met top to bottom. I tried multiple methods to fix this seam such as mirroring the UV in 2 lines of symmetry and blending flipped copies of the same noise texture, both of these methods removed the seams by left other visual anomalies. Eventually I found an open implementation of 3D simplex noise and then took noise values for each vertex by its world position coordinate.
- Created a lightning effect by emitting particles of a yellow-white light, changing their velocity randomly (to make them zig zag) and drawing a trail of where they've previously been.
- Wrote a C# script to shake the camera when the player comes into contact with the lightning to give the player visual feedback.

End of Term 1 Week 2

- Finished up the proposal and now had a solid idea of what I want to create.
- Made the movement system better by interpolating towards desired velocities, rather than just using simple impulses
- Implemented a first version of the dash mechanic, which lets you move (almost instantaneously) a certain distance in a direction.
- Created a skate fish model in blender
- Animated the skate fish using a shader which displaces vertices proportional to: distanceToCentreLine^power * Sine(time). This gives the effect of the skate flapping and curving its wings.
- Created a jellyfish model in blender
- Animated the jellyfish using a shader which displaces vertices proportional to: distanceFromHead * Sine(sumOfWorldCoords + time). This gave the effect that each tentacle was moving independently.
- Created a cliffside with the Unity terrain tools where the player is launched into the pit from at the start of the game.

End of Term 1 Week 3

- Updated the gantt chart based with the knowledge I now had about the feasibility of some of my ideas
- Added a "hurt" effect that will communicate that the player's health is low without the need for a head up display, the effect can be intensified based on how much health is missing

- Added a shield effect that gives feedback to the player when they have used a certain ability (parry)
- Added a warp speed lines effect to communicate to the player how fast they are descending
- Made a moray eel model in blender
- Animated the moral eel with a shader based on the Sine of each vertices world position, this leads to it moving through the air similar to a snake.
- Added more complexity to the tunnel shader so a greater variety of effects could be created
- Created a particle effect to hide the bottom of the tunnel in an approximation of fog (but with the ability to change the fogs colour), this way the player cannot discern that the tunnel is being created on the go as they fall.
- Replaced the default using skybox with a shader that looks like moving clouds and added a “sun” object (a glowing sphere high above)
- Added a check to make sure the player does not go outside a radius from the centre, this acts as a collision checker for the pit wall but is much cheaper to compute than my first method which simply used the section’s mesh as a collision object, however this lead to unusual behaviour where the pit sections meet and sometimes resulted in the player having friction against the walls even when the friction value was supposedly set to 0.
- Added a cloud blanket (big plane + shader) at the bottom of the cliffside
- Added fade (C# script) when falling through the cloud blanket, so the player sees a seamless transition between the cliffside intro and the pit gameplay.

End of Term 1 Week 4

- Started work on the interim report
- Started on “egg” object (made shader for inner and outer layer) that could either work as a speed boost or health pack
- Added “nibbler” obstacle visuals (pack of small bioluminescent creatures that nibble on you) as a vfx particle object
- Added toxic fog obstacles visuals with vfx particle graph.
- Made rudimentary input system (dash, tap parry, hold parry)
- Hooked up ability effects to button presses
- Hooked up health to hurt effect
- Made fade to black and reload scene on death (health < 1)
- Stop relying on rigid body drag and instead manually clamp and interpolate velocities for more movement control
- Implement first version of movement changes when holding parry
- Change speed lines from cpu particles to gpu particles for performance enhancement
- Change dash behaviour (dash velocity is constant and cannot be changed mid dash)

End of Term 1 Week 5

- Finished first draft of interim report
- Decided on trying to use orange spots for danger, blue stripes for parry
- Changed back from 1 button for parry and dash to two buttons, it just feels nice to use and doesn't lead to accidental wrong ability uses.
- Added a dash down ability.
- Added logic to eel that makes it move towards the player once they get within a certain range
- Changed eel movement shader animation so that it's more noticeable when looking down from above
- Added logic to make skate swim in a circle.
- Added logic to make lightning periodically turn on and off.
- Changed skates animation to intensify based on speed
- Fix some scaling issues I created when outputting with the wrong settings from blender
- Added rudimentary procedural generation, spawn a random obstacle in each pit section.

End of Term 1 Week 6

- Finished second draft of interim report
- Made some more complex tentacle obstacles (i.e. containing multiple tentacles moving together)
- Made more complex lightning obstacles.
- Implemented a system to transition between tunnel shader presets.
- Made a title screen
- Made a quick intro cutscene to give slight context to why you are falling through a tunnel.
- Added (possibly placeholder) textures to jelly, eel, tentacle and skate

Supervisor Log

Meeting - 13/09/19 10:00

- *Discuss ideas about the project*
- *Agree to use Trello to plan the project as well as for sharing documents, progress and feedback*
- *Go through project timescale and what needs to be submitted when*
- *Discuss common pitfalls in similar projects and how they could be avoided (e.g. “how much of this did you actually make?”)*
- *Talked about how to avoid not finishing the project for the deadline*
- *Brief look at previous prize-winning projects*

Trello - 16/09/19

- Discussed the use of visual scripting tools instead of visual ones
    - Okay as long as there is substantial written code as well
- Talk about whether I or the university own the rights to my video game
    - Didn’t know, asked academic supervisor and he thinks I own the rights
- Float the idea of making video logs

Meeting - 08/10/2019 10:00

Questions about project proposal:

- Structure and content of aims and goals
    - Required, extensions, extensions can be vague
- Structuring a Gantt chart? Develop, test, develop?
    - Discuss things I should have as tasks on my gantt chart
- What to talk about in relevance
    - Jobs, software engineering, programming languages
- Related projects, other students? Other game projects from other unis?
    - In regards to proposal, not really important
- Two pages or two sides?
    - Not important, proposal is just for supervisor to get an idea of what the project is about

Other questions:

- When to involve experts
    - Once the game is playable and would benefit from feedback
- Using industry interviews or talks as sources

    - Those kinds of sources are acceptable

Trello - 19/10/2019

- Ask about possible source
    - Not supervisors research area, reminded of types of sources we discussed before.
- Asked for feedback on gantt chart
    - Gantt chart is good, but add heuristic evaluation as a task

Meeting - 05/11/2019 10:00

- Discuss interim first draft.
- Remove narrative style, formalise
- Number sections
- Number figure captions and put below images
- Change a few words
- Clear up confusion about expert feedback
- Describe tasks in more detail
- Describe completed work in more detail
- Restructure introduction to conform to guidelines.
- Fix sentence structure

Ethical Compliance Form for UG and PGT Projects*
School of Engineering and Informatics
University of Sussex

This form should be used in conjunction with the document entitled "Research Ethics Guidance for UG and PGT Projects".

Prior to conducting your project, you and your supervisor will have discussed the ethical implications of your research. If it was determined that your proposed project would comply with **all** of the points in this form, then both you and your supervisor should complete and sign the form on page 3, and submit the signed copy with your final project report/dissertation.

If this is not the case, you should refer back to the "Research Ethics Guidance for UG and PGT Projects" document for further guidance.

---

1. Participants were not exposed to any risks greater than those encountered in their normal working life.

   Investigators have a responsibility to protect participants from physical, mental and emotional harm during the investigation. The risk of harm must be no greater than in ordinary life. Areas of potential risk that require ethical approval include, but are not limited to, investigations that require participant mobility (e.g. walking, running, use of public transport), unusual or repetitive activity or movement, physical hazards or discomfort, emotional distress, use of sensory deprivation (e.g. ear plugs or blindfolds), sensitive topics (e.g. sexual activity, drug use, political behaviour, ethnicity) or those which might induce discomfort, stress or anxiety (e.g. violent video games), bright or flashing lights, loud or disorienting noises, smell, taste, vibration, or force feedback.

2. The study materials were paper-based, or comprised software running on standard hardware.

   Participants should not be exposed to any risks associated with the use of non-standard equipment: anything other than pen-and-paper, standard PCs, mobile phones, and tablet computers is considered non-standard.

3. All participants explicitly stated that they agreed to take part, and that their data could be used in the project.

   Participants cannot take part in the study without their knowledge or consent (i.e. no covert observation). Covert observation, deception or withholding information are deemed to be high risk and require ethical approval through the relevant C-REC.

---

*This checklist was originally developed by Professor Steven Brewster at the University of Glasgow, and modified by Dr Judith Good for use at the University of Sussex with his permission.

If the results of the evaluation are likely to be used beyond the term of the project (for example, the software is to be deployed, the data is to be published or there are future secondary uses of the data), then it will be necessary to obtain signed consent from each participant. Otherwise, verbal consent is sufficient, and should be explicitly requested in the introductory script (see Appendix 1).

4. No incentives were offered to the participants.
   The payment of participants must not be used to induce them to risk harm beyond that which they risk without payment in their normal lifestyle. People volunteering to participate in research may be compensated financially e.g. for reasonable travel expenses. Payments made to individuals must not be so large as to induce individuals to risk harm beyond that which they would usually undertake.

5. No information about the evaluation or materials was intentionally withheld from the participants.
   Withholding information from participants or misleading them is unacceptable without justifiable reasons for doing so. Any projects requiring deception (for example, only telling participants of the true purpose of the study afterwards so as not to influence their behaviour) are deemed high risk and require approval from the relevant C-REC.

6. No participant was under the age of 18.
   Any studies involving children or young people are deemed to be high risk and require ethical approval through the relevant C-REC.

7. No participant had a disability or impairment that may have limited their understanding or communication or capacity to consent.
   Projects involving participants with disabilities are deemed to be high risk and require ethical approval from the relevant C-REC.

8. Neither I nor my supervisor are in a position of authority or influence over any of the participants.
   A position of authority or influence over any participant must not be allowed to pressurise participants to take part in, or remain in, any study.

9. All participants were informed that they could withdraw at any time.
   All participants have the right to withdraw at any time during the investigation. They should be told this in the introductory script (see Appendix 1).

10. All participants have been informed of my contact details, and the contact details of my supervisor.
    All participants must be able to contact the investigator and/or the supervisor after the investigation. They should be given contact details for both student and supervisor as part of the debriefing.