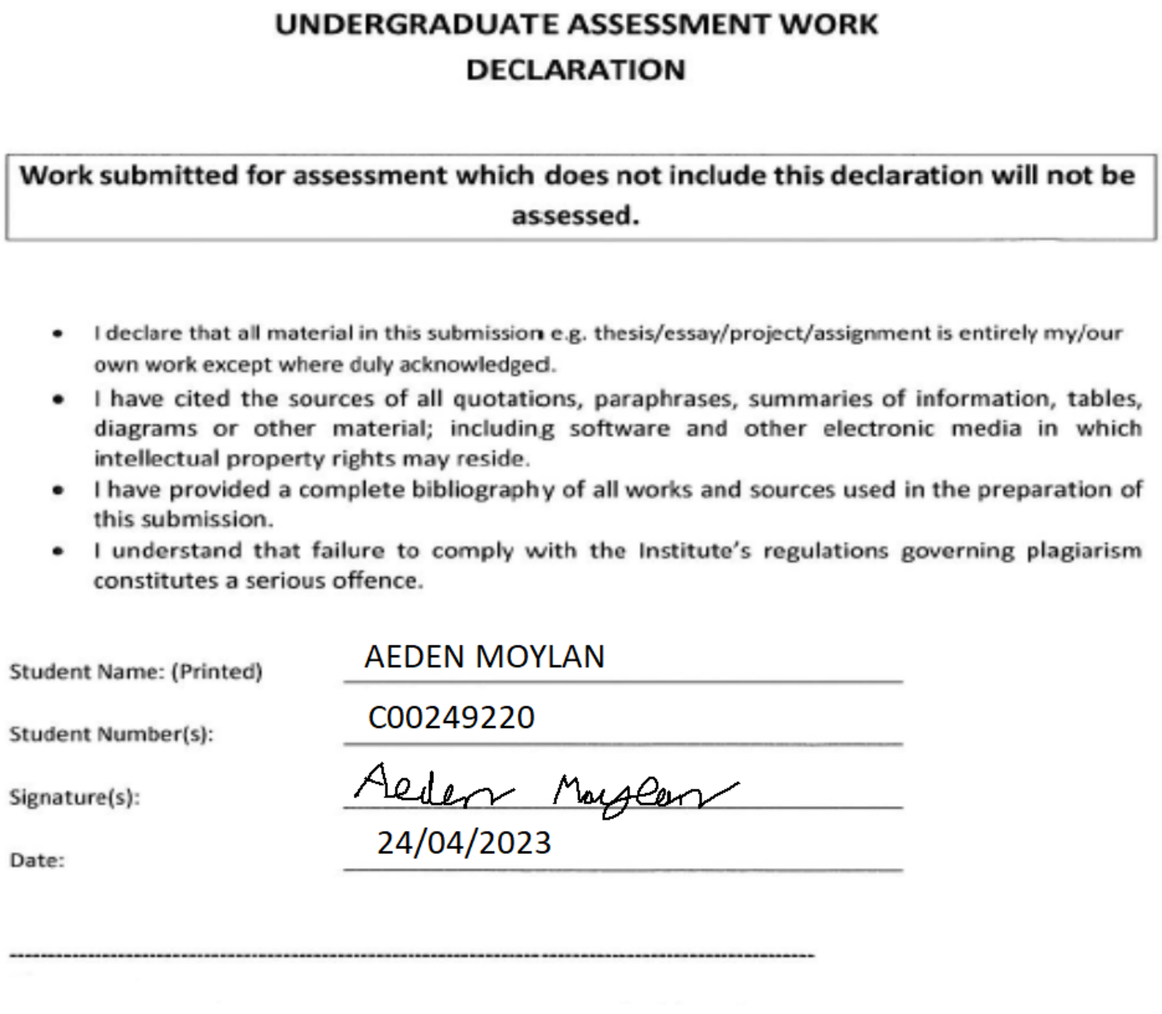
Ollscoil Teicneolaíochta an Oirdheiscirt
South East Technological University

Computer Games Development Project Report Year IV

How Easily Can Immersion In Video Games Be Achieved?

Aeden Moylan
C00249220
Date of Submission:24/04/2023
WORD COUNT:

DECLARATION



UNDERGRADUATE ASSESSMENT WORK DECLARATION

Work submitted for assessment which does not include this declaration will not be assessed.

- I declare that all material in this submission e.g. thesis/essay/project/assignment is entirely my/our own work except where duly acknowledged.
- I have cited the sources of all quotations, paraphrases, summaries of information, tables, diagrams or other material; including software and other electronic media in which intellectual property rights may reside.
- I have provided a complete bibliography of all works and sources used in the preparation of this submission.
- I understand that failure to comply with the Institute's regulations governing plagiarism constitutes a serious offence.

Student Name: (Printed)	AEDEN MOYLAN
Student Number(s):	C00249220
Signature(s):	Aeden Moylan
Date:	24/04/2023

Contents

Acknowledgements

I would like to thank the following people who assisted in completing this project. Dr Oisin Cawley acted as my supervisor for this project and provided a lot of insight and ideas for potential improvements. I would also like to thank Dr Lei Shi who is in charge of sending all relevant information about the Final Year Project as well as handling deadline dates and project presentations. I also want to thank all of the lecturers from the Department of Computing at SETU as I would not have made it to the level of knowledge and confidence that I have today without their wisdom and patience throughout the years. I would also like to thank my family for their constant support, and my fellow friends from the Game Development course who have provided a comfortable environment and laughter even at the most stressful of times.

Project Abstract

The horror genre of games has a massive focus on realism and immersion, but the genre has been vastly unexplored in terms of new ways to experience this. Games can only be so immersive when it is just a screen and a controller. This project is looking deeper into ways to improve this even more by using other parts of your body as well as using realistic dungeon-like map generation algorithms. Other features or topics to be looked at will be how spatial sound, lighting, and enemy AI can affect the atmosphere and overall feel of the game. I will expand and experiment with this idea to make an atmospheric and cohesive random map every time using a

grid-based algorithm for the map layout, and a tile-based system for decorating the map afterwards. I will also use the Tobii Pro line of eye trackers to track where the player is looking on the screen to enhance the users' experience by making it a core game mechanic. Video games have used similar mechanics such as virtual reality where the player can look in a direction and the in-game camera will follow, however, few games track where the eyes themselves are looking. The overall goal of this project is to make an immersive and scary horror game that takes advantage of these features.

Project Introduction

For my thesis Project for the completion of my Computer Games Development course, I decided to research and create a 3D horror game using a procedural map and room generation as well as eye tracking to implement a study of the features that improve the immersion of video games. The main reason for choosing this as my research question to base my project on is that I wanted to explore the possibilities and limitations of implementing eye tracking and random map generation in a horror game setting. The main game engine I used to create the game in Unity 2021.3.10. The main concepts and aspects I added to the game are as follows, an enemy with different states, animations, and AI implementation, Eye tracking using an external eye tracker, grid-based procedural map generation, tile-based room decorating, and overall gameplay additions to increase immersion.

To fully understand why and how a study of the features that improve the immersion of video games, specifically that of the horror genre is linked to my final project, I will fully define and explain the steps I took, the techniques I implemented and include the research I have completed in tandem to create the horror game.

The premise of my game is that you play as a man trapped in a house with a deadly and mysterious killer. There are no exits and the only way you can survive is to open up a safe which contains a gun. You arrive with nothing except a small flashlight which you must use to navigate the dark building and find the keys to open the safe and acquire the gun. The house and room layout are completely random each time and the location of the keys are different as well. The flashlight will shine on the screen wherever the player is looking and is the main source of light. The killer is in the house as well and will wander the halls in search of you. Wardrobes are scattered around the house that you can use to hide in. If the killer sees you, he will sprint at you and if he reaches you he will kill you. It is up to you to safely wander these halls in the hope of survival

The main inspiration I have taken for this game is that of Amnesia Collection created by Frictional Games in 2014 using the HPL 2 game engine as well as Dreadout (2014) which utilises eye tracking as a flashlight. Dreadout also uses the Unity engine so I wanted to try and replicate something similar with my own project.

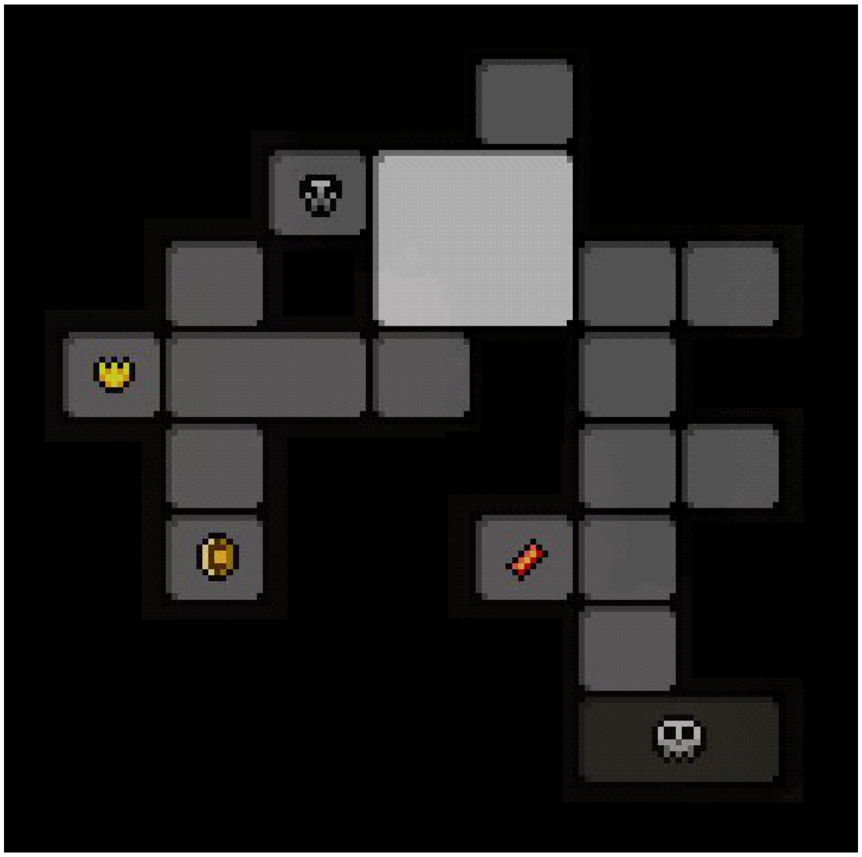
Literary Review

From my research, I have concluded that there are multiple areas that horror games focus on to massively improve immersion and scare factor. These are atmosphere, Sound design, lighting, and the feeling of uncertainty and being unpowered. I wanted to make this horror game with those factors in mind to see if the end result adhered resulted in a genuinely immersive game. Greg Kasavin, a designer for games studio SuperGiant Games stated that “The Atmosphere in games is the hidden layer between the artwork, audio, narrative, and level design”. This project will focus on many of these features starting with level design. (Kasavin, 2012)

Level Design:

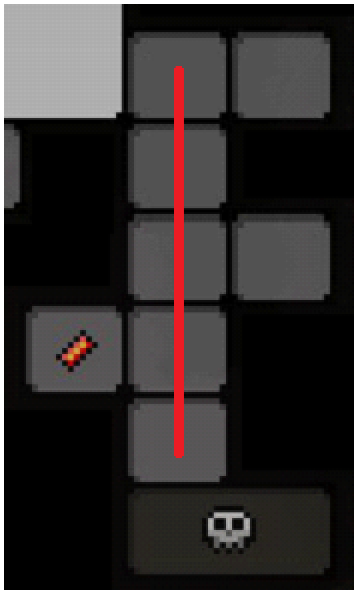
The first game with a procedurally generated dungeon-crawler was called "PEDIT5" [Rusty Rutherford]. This featured a predesigned map which means that the layout never changed but with random enemies and treasure encounters. A game called "Rogue" [Epyx] was the first game with a fully random dungeon map generator. Rogue was the stepping stone for the map generation algorithms we have today. The biggest source of inspiration in terms of the map generation algorithm is a game called "The Binding of Isaac: Rebirth" [Nicalis]. This game consists of progressing through a set of levels, each with a random dungeon map layout. An example of the map is shown below

(Boris, 2021)



This game uses a grid-based system to randomly generate a dungeon map. Each map must contain 1 of certain types of rooms such as a boss room or an item room. The room type is decided after the map itself is generated. Rooms with only 1 neighbour are marked and can be changed into a special room type like a boss room. Once the layout of the map is generated, the game randomly picks from a few premade-level designs and inserts them into the normal rooms. The algorithm I want to make is similar to this one, however, there are a few major changes.

The first change is that my algorithm will include long corridors. If there are multiple numbers of rooms on the grid that are connected, they will be assigned a corridor room type. The picture below shows an example of this. The rooms with the red line will be transformed into a long corridor. The connected rooms will be unchanged



The other massive change is how the rooms themselves are generated. Instead of picking from multiple premade room designs, the rooms themselves will be assigned a theme type. For example, one room could be assigned to be a bathroom while another could be a living room. A separate algorithm will be run to populate these rooms with the appropriate room assets. Of course, the assets have to allow full unblocked access to the room and have to make logical sense. This algorithm will differ for each type of room as each room will require different placement of assets. Keys will also be placed in random rooms.

Another game I looked at for map generation was Diablo 1. Diablo 1 has 4 levels each with Its own level-generation algorithm. Each level was built within a 40 x 40 grid of tiles with each tile representing a room on the map. The majority of the rooms are not made using premade room assets which means that the majority of the rooms are procedurally generated at runtime. Per level, there are a few key rooms that must appear on each level in order to progress. These important rooms are in fact premade chunks that are then inserted onto the map after it is made. They also have what are called “Theme Rooms”. These rooms have a unique theme to them which changes what kinds of items will be placed inside of them. For example, libraries have to spawn bookshelves and monster pits must spawn monsters. All of this information was taken from a game developer’s conference (Brevik, 2016)in which David Brevik gave an in-depth speech about the inner workings of Diablo.

The big question is how can we populate the rooms. There are 3 standard ways to do this according to archmagerises.com. These are using chunks, chunks with tiles, and pieces (Henshell, 2021).

chunks involve a level designer making a chunk of level geometry. After the fact, another person manually inserts the assets such as tables or chairs into the level. This makes a fully designed chunk that can be dropped into any part of the grid. This way is not random and therefore does not suit my purpose.

chunks with tiles involve a similar approach. Level designers design a chunk of level geometry. The difference is that the assets are placed at runtime using an algorithm rather than at design time using humans. These chunks may not be close to each other so tiles of floor and wall are used to fit all of the pieces together. The algorithm more or less fills the gaps. This is closer to what I want, however, my level design will be made on a 2D grid where every room is connected so there is no need to fill in the gaps as there won't be any. This type of level design algorithm is used a lot with top-down dungeon crawlers such as Diablo 3 [Diablo 3]. Since my game will be first person there is even less need to fill in the gaps as the player won't be able to see beyond their field of vision.

The third method is what I believe is most similar to what I want. using the pieces method each room splits into a smaller grid. For example, a room could be split into a 10 x 10 set of tiles. the walls and corners are then placed on the appropriate tiles to form the room's perimeter. Then the remaining tiles will be populated with a floor asset. Other assets can be placed on top of the floor tile by tile. One tile could place a table and another could place a lamp. The assets cannot be placed in the same tile. There might be types of assets that are allowed to be placed on top of other assets such as a cup on a table. The table and the lamp will be marked as core assets in the room as it contains collision detection so that the player cannot walk through them. The cup however will be marked as a decoration and will not hinder the player's movement. The issue is that the tile method only allows you to place objects that are within the size of the 1 tile.

I believe the tile method would work best for me with a few changes. The biggest hindrance is the 1 x 1 tile size restriction so placing other objects might be a challenge. I believe it is possible however to have large objects cover multiple tiles. E.g. a table might cover the size of 4 tiles in a 2 x 2 area. Those tiles will not be allowed to place a core asset on top of it but decoration assets will be allowed to.

The tile

With the map generation sorted, I did research into the best way to decorate the rooms dynamically. I looked at how other games were populating their rooms with appropriate assets while also making it look natural.

Lighting:

Effective lighting is a crucial aspect of horror games, as it has the power to make an environment feel completely different while not changing the base design of the map. (Pluralsight, 2017) has a great article on the subject that adding a definitive source for your light source is an important step in establishing light in your game. If the light comes from an unknown source, It can act as a way for the player to disconnect from the feel of the game. After reading this I researched how other games in the horror genre approached lighting and looked to see if they adhered to this philosophy as well. One of the researched games is a game called “The Last of Us” [Naughty Dog]. This game features dark sections where the main character must rely on a small flashlight that always points in the direction they are facing. This makes the definitive source of the light attached to the player itself. This inspired me to use the idea of the player as the main source of light in this game. However, “The Last of Us” is a third-person game and I wanted my game to be first person so I did more research on how I could make the player be an effective source of light in a first-person view.

A first-person horror game by the name of “Amnesia: The Dark Descent” uses the player as the main source of light in a lot of situations as well. The player can take out a lantern that emits light in a dark room. The lantern has a limited amount of oil so it is meant to be used sparingly. The only other main sources of light come from lightable candles on the wall which need a tinderbox to light. These tinderboxes are also limited in supply. The limited light sources, flickering candles, and shadows all work together to create a sense of fear and danger, adding to the overall horror experience of the game. Using this game as a reference, I wanted the player’s light source to also have some form of restrictive usage as having a light always turned on would remove some sense of danger. I also liked the idea of the old-fashioned candles on the wall as the only other source of light.

I wanted the light in the game to be an interactive feature that the player could use and control. Amnesia simply had a toggle to turn the lamp on and off but I wanted more control of the light source. This research led me to the idea of using eye tracking to control a source of light from the player.

The final game I found for my lighting research is a game called “Dreadout”. This is a third-person horror where you play as a high school student trapped in an old abandoned town. This game uses eye tracking to control both a flashlight that the player holds and a screen that can expand with your gaze. This game proved to be the biggest inspiration for my project as I loved the idea of controlling a flashlight with eye movements. I combined the eye-tracking capabilities of the flashlight from Dreadout with the first-person view, and limited usage of the lamp from Amnesia to create the main source of light that I wanted in my game.

Eye Tracking:

Implementing the eye tracker element of the game was difficult at first. The tracker I used to implement it was Tobii Pro Nano. It had a difficult time recognising the device, and my pc and didn’t connect well. To solve this, I consulted troubleshooting options on both Unity and the Tobii site as well as contacted the Tobii customer support team. The Tobii customer team sent me access to a Tobii pro package that I could import to Unity and after that, it worked as expected, I realised at this point too that in order for the tracker to work efficiently and to the best of its ability, it works better on a smaller monitor as the max supported screen size of the Tobii Pro Nano is only 24”. This eliminated my 32-inch monitor from being usable in this project. The model of the eye tracker is one of the main faults I made throughout this project. I used my college-issued PC to complete most of the programming for this as the screen size was within the 24” limits. I wanted to ensure that I got as accurate eye movements as possible so to research this I read a paper by Limin Zhang and Hong Cui, The main focus of their paper was to investigate the “Reliability of MUSE 2 and Tobii Pro Nano at capturing mobile application users' real-time cognitive workload changes”(Zhang, Cui, 2022).

Whilst my game is not a mobile one, it contained valuable findings in regard to capturing accurate calibration in relation to eye movements. I implemented it by adjusting my seat to a slightly higher level to be as in line with the tracker and screen as possible, just like what was mentioned in the paper. “ we had a higher chair for the participants to improve the capture rate and adjusted the Tobii Pro Nano angle according to each participant.”(Zhang, Cui,2022). As it was just me testing the tracker initially, I had no data to compare efficiency rates to so once I got it working smoothly I kept the settings as they were.

I had to download various updates of the Tobii Eye tracker application to get it to actually function initially. It is using a Tobii pro package designed for Unity which contains all the scripts needed. I remade a few of them for finding any connected eye tracker and connecting it to the game. I did this to ensure that regardless of what PC I used be it my at-home set-up or college one, I would be able to use the eye tracker regardless of location. One script I did not change basically calculates the information needed which I thought wouldn’t be needed to be changed as it functioned effectively within Unity. It also calculates information such as whether the eyes are open or whether the eyes are looking at the screen or not. It was crucial that I got this working effectively as early as possible as for me this was a key element I wanted to include in the gameplay. I wrote a script that accesses this information and gets information from it to determine where the coordinates of your eyes are looking on the screen. I added this in addition to programming a unity light to take said coordinates and once the coordinates are found, a ray cast is shot out from them and the light is pointed to whatever object the gaze hits. There is also a check to see how many eyes the user has open. If 2 eyes are open it will shine the light wherever they are looking. If there is only 1 eye open the gun the player can carry starts to move until it is positioned in a way that aims down the sights.

I feel that these features really enhance the player's experience in the game by giving another part of their body a very important gameplay function that affects the gameplay. Overall it gives the player a feeling of being more immersed in the virtual game.

Enemy:

In regard to enemy implementation, there were many factors I had to consider. These included the enemy Ai algorithm, different states, and how the enemy would interact with the player. One of the most influential horror game enemies comes from a game called “Amnesia: the dark descent”. In this game, there are enemies called “The gatherers” (Amnesia Wiki,2023). These enemies wander the halls and corridors looking for the player as they progress through the game. The enemies aren’t very clever as they can easily lose sight of the player as they hide behind obstacles or inside wardrobes. In regard to the enemy in my game, a lot of the features I have mentioned above have also been implemented. The enemy has multiple states starting with the “Wander” state. In this state, he wanders similarly to the gatherers as mentioned previously. He picks a random room in the generated map and starts to walk towards it. If he reaches the room in this current state, he will change to a “Looking” state. In this state, he looks all around him in order to try to find the player. If he cannot find you, he will start to head to another room and repeat this.

I wanted to research different ways the enemy could detect the player. I found a game called “Shadow Tactics” that added a vision cone system for the enemies to detect the player. In this game, the enemy's line of sight is a critical element that players need to consider when planning and executing their moves. The enemy line of sight in the game is represented by a cone-shaped field of vision that extends from the enemy. The enemy's line of sight also takes into account obstacles such as walls, bushes, and other environmental features. If a part of the vision cone hits an obstacle or wall, the enemy line of sight gets cut off from viewing what is behind it. The vision cone is more in-depth than this, but these are the features from it that I wanted to implement in my game. (Game DeveloperStaff, February 13, 2017). The vision cone was implemented in my game by making a script that makes a lot of triangles at runtime that extends in front of the killer in an arc shape. A ray cast is then shot out in the direction that the triangles are facing. If the ray cast hits an obstacle or wall, that specific triangle will only extend the distance that the ray cast reached. This means that if the player is hiding behind a wall or obstacle, the vision cone cuts off in sections and the player cannot be seen. This helps with the immersion of the game as the killer has a fair and accurate way of spotting you based on what walls and obstacles are nearby.

I also wanted the killer to be able to place some sort of trap that would hinder the player’s progress of the building. In my research, I came across a game called “Dead by Daylight”. This is a 1v4 game where a killer has to find and kill a group of 4 survivors while the survivors have to escape. There is a certain killer from this game that caught my eye called the trapper. The trapper can place down beartraps in strategic places in order to catch the survivors off guard (Behavior Interactive, 2016). My killer uses this idea to place a beartrap at the front of a wardrobe if he sees you entering one. The player will be forced to leave the wardrobe in which case they will get trapped in the bear trap. If the player enters a wardrobe and is not in range of the killer’s vision, the killer will simply walk past the wardrobe until it reaches the end of the corridor. I wanted to add this to create a sense of uneasiness for hiding in wardrobes. Wardrobes are the only safe hiding spots on the map and adding this feature adds a bit of risk in using them.

Evaluation and Discussion

The purpose of this project was to create a game with the focus being to make it as immersive as possible using methods I have researched. Immersion is something that cannot be specifically measured with numbers as it changes depending on the person playing the game. One person might be absolutely terrified while playing the game, while other people simply do not find horror games scary. A study was done that compared the experience that several different types of gamers had while playing the horror game “Alan Wake” (Windels, 2011). The experience and perception of the game at key moments had a different impact on each person depending on whether they were casual or experienced gamers. Several different biometric tests were being monitored on each individual that were tracking things such as heart rate, Skin surface temperature, and Galvanic skin response. These were used to calculate how the player’s body was reacting while playing the game. For example, an increase in heart rate is expected if the player is feeling frightened. The study showed that casual gamers were scared more often in situations that experienced gamers were not while playing this game. Using this information it is hard to grasp just how scary my game is as the results could differ a lot depending on the person playing. I decided that the best way to measure if my game was immersive was to focus on the elements that I researched that make games immersive and then compare those elements that I added with how other games in the same horror genre implemented them. I wanted to replicate how these features added to the overall feeling of the game. I felt my game was complete when I saw similarities in these features. When I was done adding these features, I noticed that my heart rate was higher as I was playing the game and after I was done playing I was a little bit more jumpy and anxious.

Project Milestones

Probably the biggest project milestone in my project was making the grid-based procedural map algorithm. It took me a few attempts to get it right and once I did I felt like this aspect really added to the game. It prevents the game from becoming boring too quickly thus creating an engaging and fresh experience for the player should they choose to play more than once. I felt this was crucial as I feel that this element is an important factor in ensuring that the gameplay is immersive and not too repetitive.

the tile-based room generation and decoration was another milestone I was proud to reach. For the majority of the project, I was programming an environment that was very flat looking and was simply, just grey walls. When I reached the point of decorating the rooms by filling them with assets etc I was able to feel more in the game’s environment and it felt more realistic and put together. These milestones were the essential ones I wanted to hit before the deadline implementing the eye tracking correctly. I was having difficulty getting it to work correctly in the beginning and I feel that once I achieved this, I reached a major point in my game's creation. At the start, I mentioned in my Introduction that I feel that eye tracking is an effective immersion tool in gameplay. That is why it was a major marker to implement this element into the game to make it as immersive and interactive as I possibly could.

I was also happy with how the enemy turned out. He started off as a simple Unity store asset, but as time went on and he was fleshed out with AI, States, and Animations, he was a lot more unique. He was implemented in a way that made him constantly put the player at risk in one way or another by constantly walking through the corridors or rushing to your position after stepping in a bear trap. He became a real asset to the feel of the game and contributes a lot to the atmosphere provided.

Major Technical Achievements

The main focus of this project was to improve immersion in a 3D horror video game. I believe this was achieved using the researched points of what makes a video game immersive. These include level design, lighting, and sound. The final project represents a combination of these things in unison with my own ideas.

The Grid-based map generation and the tile-based room generation were probably the most technical part of this project. Generating a completely random map layout without any overlapping or logical inconsistencies was a challenge in and of itself. However, the map was then populated with appropriate template rooms to represent what room type was needed. For example, a curved corridor room asset would have to be placed where the map wants to change direction, or an end corridor would have to be placed in an area where the corridor ends. All of these things combined made this a challenge in and of itself. These rooms are then populated using a tile-based map decoration system. This also has a lot of rules and checks such as certain objects can only be placed adjacent to visible walls, or some objects cannot be placed at all. This also has to make logical sense and also has to be traversable. Once that was done non-obstacle decorations could be placed to make the game feel more atmospheric and natural. The result is a fully procedurally generated map with fully dynamic decorations and obstacles that are completely different for every playthrough.

Project Review

As I am coming close to the end of my project, I cannot help but reflect on the progress I have for the past few months. I am happy with the outcome of the project but I do have a few comments on the different stages of this project. I started this project by designing the map generation algorithm and that alone took more time than I was expecting. Each room must connect with one another in a way that makes logical sense and is completely random every time. I learned it is very hard to replicate logical environments that are randomly generated, and in order to make them work the algorithms must be fine-tuned to a massive degree so that no small inconsistency remains. It is like asking artificial intelligence to make a 3D sculpture while also expecting it to look beautiful to the human eye. Once the map layout algorithm was designed, I then moved to the key gameplay features. I believe this was the right choice as a game is nothing if it cannot even be played. After these were done I went back to adding features to boost immersion such as Lighting, Audio, and Room Decoration. I am very happy with the Lighting and audio as I feel it adds a cold feeling to the game which is what I was going for. The room decoration was equally as difficult as the map generation in my opinion. Decorating a room has more of a focus on being appealing to the human eye than a random layout. The algorithm must decorate the room in a logical way while also being traversable. This means that even more fine checks are needed in the algorithm to ensure that moving through a room is even possible. These 2 algorithms took up a major part of the time overall on this project. If anyone else is doing a similar project I would recommend triple checking if a dynamic map is needed as a lot of your time and effort will go towards it. One issue I had with my game was to do with performance. Turns out that having a large number of rooms that are populated with a large number of assets does not help with how well your game runs. It took a lot of cleaning and refactoring of my project to try and remove as many unnecessary assets as possible in a bid to improve performance and although it did help, my game won't get over 120 frames per second. I'd recommend doing more research on how to improve performance despite having hundreds of different assets in the level at a time. Maybe investigate ways to stop rendering an object if it is not in your line of sight. Unity was the perfect game engine for my needs. It supported the Tobii Pro line of eye trackers and really assisted in my goal of increasing immersion using a couple of prebuilt features such as their excellent lighting system, Audio system, and was able to support the large number of objects I was generating.

Conclusions

To conclude, enemy implementation, Eye tracking using Tobii Pro hardware, Map Generation, Lighting, Audio and finally Room Decoration are aspects I included in my final year project to implement features that improve the immersion of video games, specifically that of a horror game. As a result of completing my research and finishing my Final Year Project, I have a more rounded and comprehensive understanding of what it takes to make a horror game immersive, and enjoyable with the aspect of fear.

Outlining my overall project abstract, idea, key concepts, research, milestones, results and general reflection helped me to reach the conclusion that overall the more immersive the gameplay the more enjoyable the player experience is from my time playing the game I created as well as getting others to try it out for themselves.

I have always had a passion to find out what made games so atmospheric and enjoyable. Sometimes the reason was because of an engaging story, or maybe jaw-dropping landscapes and environments. For Horror games, the reason is they trigger one of the most primitive emotions that humans have, Fear. This fear is what drives players to horror games as it offers a way to feel afraid in a safe and controllable environment. It also provides a rush of adrenaline which can make the experience even more enjoyable.

This led me to the idea of using eye tracking as a core game mechanic as it forces you to focus on the screen which immerses them in the game more and deprives them of the option to simply look away. Examining new ways to make the player itself a part of the game is a fun and exciting venture for the future game industry as technology advances. This is why I feel that this genre of game was the best in examining how different features when used together can contribute positively to the player experience.

According to Kane Williams, “If a game is immersive and well-realized, the player can completely lose themselves in play, allowing everything to feel more real. This sense of false realism is what makes the scares that much more effective.” (Williams,2022). I feel that in essence that was my main objective and what I set out to achieve whilst creating this game.

References

[Rusty Rutherford] PEDIT5[Video Game]. (1975) University of Illinois.

[Epyx] Rogue[Video Game]. (1980) California.

[Nicalis] The Binding of Isaac: Rebirth[Video Game]. (2014)

|Boris (2021) *Dungeon generation in binding of Isaac*, *BorisTheBrave.Com*. Boris. Available at: https://www.boristhebrave.com/2020/09/12/dungeon-generation-in-binding-of-isaac/ (Accessed: May 2, 2023).

Henshell, T. (2021) *How to procedurally generate and decorate 3D Dungeon Rooms in Unity C#*, *Archmage Rises*. Archmage Rises. Available at: http://www.archmagerises.com/news/2021/6/12/how-to-procedurally-generate-and-decorate-3d-dungeon-rooms-in-unity-c (Accessed: May 1, 2023).

[Blizzard] Diablo 3[Video Game]. 2012

Coordinate systems - Tobii Pro SDK documentation (no date). Available at: https://developer.tobiipro.com/commonconcepts/coordinatesystems.html (Accessed: May 1, 2023).

[Digital Happiness] Dreadout[Video Game]. (2014) Indonesia.

Kasavin, G. (2012) *Creating atmosphere in games*, *GDC Vault*. Available at: https://www.gdcvault.com/play/1015556/Creating-Atmosphere-in (Accessed: May 1, 2023).

Light up your world: How lighting makes all the difference for games (2017) *Pluralsight*. Available at: https://www.pluralsight.com/blog/film-games/understanding-the-importance-of-lighting-for-games#:~:text=It%20can%20make%20a%20place,look%20at%20a%20horror%20game (Accessed: May 1, 2023).

[Naughty Dog] The Last of Us[Video Game]. (2013) California

'Diablo': A Classic Game Postmortem (2016) *YouTube*. Game Developers Conference. Available at: https://youtu.be/VscdPA6sUkc (Accessed: May 1, 2023).

Williams, K. (2022) What makes a good horror video game?, ScreenRant. ScreenRant. Available at: https://screenrant.com/what-makes-good-horror-video-game/#:~:text=If%20a%20game%20is%20immersive,the%20graphics%2C%20and%20the%20narrative. (Accessed: April 24, 2023).

Sanders, J. (2022) A unity review: Pros and cons, CitrusBits. CitrusBits. Available at: https://citrusbits.com/a-unity-review-pros-and-cons/ (Accessed: April 24, 2023).

Zhang, L. and Cui, H. (2022) Reliability of muse 2 and Tobii Pro Nano at capturing Mobile Application Users' real-time cognitive workload changes, Frontiers. Frontiers. Available at: https://www.frontiersin.org/articles/10.3389/fnins.2022.1011475/full (Accessed: April 24, 2023).

Behavior Interactive (2016) Dead by Daylight [Video Game]. Canada

Gatherers (2023) *Amnesia Wiki*. Amnesia Wiki. Available at: https://amnesia.fandom.com/wiki/Gatherers (Accessed: April 24, 2023).

Game DeveloperStaff
February 13, 2017 (2017) *Game design deep dive: Dynamic detection in shadow tactics*, *Game Developer*. Available at: https://www.gamedeveloper.com/design/game-design-deep-dive-dynamic-detection-in-i-shadow-tactics-i- (Accessed: April 24, 2023).

Windels, J. (2011) *Scary game findings: A study of horror games and their players*, *Game Developer*. Available at: https://www.gamedeveloper.com/design/scary-game-findings-a-study-of-horror-games-and-their-players (Accessed: April 28, 2023).