COMP30019 Semester 2 - Graphics and Interaction

Project 2: Save The Prince (Report)

Created by Kenny Lee (801361), Xin Wei Ding (758966)

Game Explanation

Save The Prince is a 3D Combat-Action driven Role-Playing Game that is heavily influenced by modern MMO contemporaries. The game follows the story of Ellen, a brave female agent who journeys her way to the faraway island of Abobora in order to complete her mission tasked by the King himself to save the Crown Prince. Upon landing on the island, she realises that there were many mystical inhabitants on the island . She will not stop to uphold her duty to rescue the Prince, even if it means slaying every Dragon and Monsters that stands in her way.

User Interface

Right from the start, we wanted our users to interact with Save The Prince like popular RPG titles such as Fortnite and GTA. Since most gamers are familiar with the interface, this would allow them to pick up the game with minimal physical interference.

Firstly, we prioritised the W, A, S, D keys to be in charge of the player's frontal and lateral movements. Then, we took advantage of the Mouse to control the camera's field of view. Also, we reserved the spacebar to handle character's jumping action. As for the special skills that we have instilled into the game, we relied on the neighbouring keys such as Q,E, and R. Lastly, we used the mouse's right and left click to handle the player's primary and secondary actions, which includes attacking.

Heads Up Display (HUD)

When it comes to designing the HUD, we had one goal in mind and that is to implement it without robbing much of the screen's real estate. From our past experience of playing MMO (Dragon Nest, World of Warcraft), we knew exactly how overloading the screen with statuses could potentially rob away the player's experience of actual gameplay. Hence, we decided to only retain information that is essential to the player on the screen.

In Save The Prince, we purposefully locate the Health Bar at the bottom left of the screen. From this position, the health status could easily be glanced over when the player needs to. Also, the health bar is sized to a length that could easily divulge critical information (character's receiving ample damage), such that it doesn't take up much of the screen. For the special skill sets that we have customised, we landed it on the opposite side of the health bar (bottom right of the screen). They are displayed as icons which indicates what type of skills that the character might display and the accompanying keys that initiates it. Lastly, when facing a monster that needs to be slayed, their health bar would present themselves on top of the screen. This would only show when the player comes into contact with a monster and would instantly drop off when either the monster dies or when the player is far away enough.

Modelling Objects and Entities

In order to portray an open world fantasy, we have utilised a vast array of models that have been taken directly from the internet. Firstly, we started our project with 3DGameKit, a starter package that is provided for free by Unity. We took in a handful of assets such as Trees, Rocks, and Cliffs in order to make our world a bit more colourful. We also borrowed Ellen, the main character of Save The Prince from the kit, seeing as she had already a bunch of animations applied, which made it easier for us to focus on other aspects of the game.

Later on, we scoured the internet to locate housing models that would populate our world. We used Unity's store, Reddit and past RPG games, and borrowed their Castles, Houses, Huts and Taverns and fix it on to our world. Lastly, all of the monsters (Dragon and Phoenix) were taken directly from the game Dragon Nest. Since all of the assets had preset animations, we took advantage of this and focused solely in managing the actions of the monsters and customising the skills that they produced.

Once all of the objects were pulled in, we designed our world accordingly to how we had envisioned it beforehand; an island that is surrounded by water and populated with a wildlife ecosystem that had traces of civilisation. All of the objects were grouped accordingly. For instance, anything that is bound to the terrain would be a child to Terrain. All of the trees that runs amok on the island is grouped together. Standalone objects are

those that have distinct traits and behaviour such as Ellen, Castles, Ships, Dragon, Phoenix and Prince.

Camera Motion

We have instilled a Third Person Camera to suite the narrative that we are trying to tell. We wanted to showcase all the interaction the main character does during combat, hence we decided to have the camera follow at a fixed distance from behind. The camera also follows the direction the character is currently facing, which eases the transition between camera movements and player movements. We limited the camera's range in terms of the Y axis, but allowed free rotation in the x axis. The camera also follows the character as she jumps and trips.

Handling Graphics Pipeline

A large part of the game that is either currently not visible or away at a fixed distance is culled in order to reduce the load at the rendering pipeline. Since the use of real-time lights, shadows and reflections impacted our game heavily in terms of performance, we used baking to precompute the lighting for our scene. Also, since we implemented shadows in to our scene especially for Trees, we tweaked the shadow distance property to only allow nearby objects cast shadows. However, since a major load of our performance issues was

due to the sheer volume of triangles that had to be calculated, we decided to limit cut down on our use of objects. And even if we had to use them, we decided to implement sparingly halving the level of details from before. All of these technique used greatly improved our game without causing any substantial lag that would hinder performance.

Shaders

Throughout the game, we have implemented 3 different types of shaders; cel shader, glowing shader, and standard shader. All of these shaders were used to enhance the user's experience when interacting with the environment.

Due to the sheer size of the island, and the lack of a mini map, we have implemented an arrow system which would guide the user to their intent destination. These arrows were implemented in such a way that they would stand out on their own. This distinction needs to clear since we want the user's to know that they are not part of environment. In order to do so, we have build a cel shader that would reflect just that.

The cel shader has four main properties; unlit color (the shadow), diffuse material color (the color of the object when hit by a light source), specular material color (the shiny reflection of the light), and the outline thickness that gives the overall object a cartoony look. The vertex shader takes care of getting the normal direction and transforming it to world space. It also calculates the light direction by interpolating the world space position, the fragment light source and the world space light source position. The fragment shader then instructs how light should interact with the object. It measures the diffuse cutoff, specular cutoff, outline strength and apply them appropriately to the ambient light and diffuse reflection and specular reflection.

Despite all the materials of rocks and mountains, the island still looked a little bland. To spice things up a little, we decided to compose a special type of rock that would only exist in the island. Glowing rocks and certain corners of the topology would bring a sort of mysticism to the land, adding a certain flair that is both visually appealing and noticeable.

The glowing shader is a simple shader that effectively enhances the visual acuity. To build our glowing shader, we decided to use a surface shader. The surface shader takes in parameters such as the color tint, the rim color and the rim power that would act as a variable, controlling the resulting glow of the object. Once the type of glow has been determined, we use the Surface Output's emission to display the actual glow.

Game Evaluation

After building up our game to a sufficient level that allows users to test out the environment and interact with the monsters in a combat setting, we decided to evaluate our game by means of both querying and observational methods. Porting the game to our laptops, we went around Baillieu Library and held our evaluation with a willing tester in a one on one setting. The testers that we found from Baillieu Library were the perfect demographic since most of them are of young adult age who at least had played some games in their past time. The evaluation was done with 10 testers over the course of 2 days. Although there were few outliers, the majority of testers were between the age of 18 - 22.

On day 1, we decided to only use observational methods to evaluate our game. We listed down a set of tasks that we wished the testers to complete. We also told them to figure out the solution by themselves and also to speak out what they are currently thinking. We then sat beside them and took down notes on how they arrive to their decision and checked if they were successful in accomplishing their given task.

The method Think Aloud was largely successful as it allowed us to peer into the users' thought process. Unfortunately, it was also during this time that we found out that there were many inconveniences and lack of key features in our game that made the game a bad experience. When the testers were instructed to perform the task of navigating the character around the environment, a handful of them were confused as to where to go. There was no minimap and there weren't any specific indicator. This forced them to rely on the footpath. As we were building our terrain, the footpath wasn't specifically designed to lead players to their destination but was just part of the aesthetic. Because of this, all of the testers were confused and would look at me and ask "Where should I go?" at least once. Since I couldn't help them and interfere with the evaluation, it took them awhile before they got to their destination.

To rectify this, we decided to build up an arrow system that would serve as an indicator for the user's to navigate throughout the island. As mentioned in shader, we purposefully created an arrow that greatly contrast the overall environment in order to grab the user's attention.

Another task that the testers had great difficulty in figuring out was displaying all of the character's skills. All of the testers were unsure as to how many skills there were and which buttons was needed to be pressed in order to initiate them. They tested it out by randomly pressing the keys on the keyboard until an action was initiated. There was confusion, because there was no indicator whatsoever that notified the users beforehand as to how to accomplish the task. As we were focused on working out the gameplay mechanics, we had entirely forgotten the need for a skill bar at the HUD. This was quickly fixed by erecting a pictured icon toolbar accompanied with their corresponding keys at the bottom right of the screen.

While exploring the environment and trying to accomplish the task that was given to them, a majority of them was trying to interact with a static objects that was not meant to be interacted with. The buildings on the island were put in place for purely aesthetic purposes, however this greatly distracted the user's in accomplishing their main task as they would try to explore it. For instance, a tester was spending a certain amount of time trying to open a door to a building that has nothing inside. Also, since the island is a big place, we have put in place a checkpoint in the form of a unicorn. This would allow the user to return to that specific destination if they were to die in combat or otherwise. However, there was a tester who kept on insisting in attacking the unicorns despite having no effect at all.

To prevent confusion from static objects and objects that were meant to interact with, we have decided to put a header that describes their purpose above the object. As for the static building, we decided to just leave it alone for the users to figure out.

One of our pride in our game was that the action combat you get to experience by fighting a big monster just like the popular title Monster Hunter. However this quickly falters, as when the testers were tasked with defeating a monster, the gameplay they had experience was suboptimal. The lag was unbearable. The frame rates were hovering between 5 - 10 FPS. Although there were some testers who was able to defeat the monster, most of them displayed frustration throughout. Some even complained and refused to finish the task. It turns out that because the machine that was running the game during testing was vastly different to the ones that we used during development, we didn't expect the toll the testing machine took after running it.

To fix this, we decided to lower the quality settings of the game. For instance, in terrain generation, we halved the amount of triangles it took to build it. We also culled the Trees that was not in view, and any far away objects. Instead of having a really big water surrounding the island, we decided to enclose the island environment with a sky cloud texture. This gives the impression of an open environment without the need of any expensive use of objects.

One day 2, we decided to use query technique in order to evaluate our new and improved game. We would allow the testers to play around the game for approximately 10 minutes before we interview them with a list of already-prepared questions. Most of the questions were inspired by day 1's feedback. Since we fixed it up, we decided to see if the testers were having any similar issues that was previously brought up. We also put in open ended questions that allowed the testers to give us their opinions on our game. Their responses were then noted down for each answer.

Questions that was asked during the interview
Did you enjoy playing the game?
Did you have any trouble engaging the user interface?
Did you have trouble using the skills properly?
Did you understand what you were suppose to do?
Were you able to navigate yourself around the island?
Did anything strike out as confusing to you?
How did you find the combat action against the monsters?
Was the difficulty in defeating the monsters appropriate? Or was it too challenging or too easy?
Did you know what the objective of the game was?

After fixing up the confusion that largely present in day 1, the opinions of the testers on day 2 were much better, especially in regards to the issues with day 1, such as "Did you have

trouble engaging with the user interface?", "Did you have trouble using the skills properly?" and "Were you able to navigate yourself around the island?". However, when an open ended questions such as "Did you enjoy playing the game?" and "Did you know what the objective of the game was?", the responses were largely mixed. Some testers knew what to do in order to complete the game while there were others who had no clue and didn't seem to enjoy it. When asked how they felt about the combat action, their responses were in the middle. They didn't seem particularly excited about nor were they disappointed by it. This could be explained by their initial expectations of our work.

Upon starting the game, we wanted the users to get excited with the combat action that was present in the game, however the reception did not reciprocate the same amount of excitement as we did. In response to this, we decided to shift our focus in creating the most aesthetically pleasing environment that the players could explore in. As mentioned above, we decided to use glowing rocks to spice up the routine. We also used implemented our water to reflect like it was actually moving. We also put up a fountain that would shoot out miasma at certain locations.

The reason why we decided to split up our evaluation strategy into 2 days was so that we could have a down time where we could fix up anything that was getting in the way of experiencing the actual game itself. When day 2 comes, we could then focus our efforts in making it to a game that was worth playing. Although the responses weren't stellar, we learnt a great deal of techniques in utilising Unity in creating a game.

Other Sources (Code/API)

A large amount of leg work such as the main character and a majority of the assets was taken directly from 3D Game Kit (https://unity3d.com/learn/tutorials/s/3d-game-kit).

As for completing the shaders, most of them were completed via YouTube as a learning source. For instance, cel shader was largely taught by (https://www.youtube.com/watch?v=3qBDTh9zWrQ), and glowing surface shader was taken from tutorials at workshops.

Camera collision utilising double sphere cast collision was taken directly from https://github.com/Datedsandwich/third-person-camera/blob/master/Assets/Scripts/CameraMovement.cs.