**TEMASEK POLYTECHNIC**

**SCHOOL OF INFORMATICS & IT**

**DIPLOMA IN GAME DESIGN & DEVELOPMENT**

**AY2023/2024 OCTOBER SEMESTER (LEVEL 2) TERM A**

**GAME MATH AND PHYSICS (CGE2C15) TERM A**

**Project Documentation (10%)**

|  |  |
| --- | --- |
| **Class** | **PC03** |
| **Name as in register** | **Yak Le Yang Ryan** |
| **Group #** | **Group 5** |
| **Topic covered** | **Particle System** |
| **YouTube URL** | Make sure the link is shareable! |

**You must submit:**

* **A discussion about the** **Technical Problems & Solutions encountered**
* **A reflection about this assignment, and also of GMAPS.**

**Each question is on a separate page below.**

1. **Technical Problems & Solutions**

**Discuss the *technical* problems you faced while writing your code, and how you overcame them.**

**Technical problems do NOT include problems you might have had with time-management, knowledge of basic GMAPS topics, illnesses, part-time work, broken computers, lost files, or other distractions, etc.**

**Focus on**

* **Problems you had in understanding concepts related to the chosen topic, and how you managed to gain a better understanding, e.g., via online references, use of ChatGPT and other AI, etc. Make sure you list all references used.**
* **Problems you had in coding your demo implementation, and how you managed to overcome these. Give specific examples.**

Your answer here (minimum 200 words for a pass grade) …

There were many technical issues I faced when running my code. One of them was that my game objects initially was flying to the sky when I tried to implement gravity on to the objects. When I set the gravity to be enabled, my objects would fly up instead of being dragged down. At first, I thought it was because of my rotation of the objects but after messing with the rotation and spawn rotation as well, it still did not solve my issue. I went online to search if others had the same issue, but I could not find any useful sources or related problems on stack overflow and other online forums. After tinkering with my particle spawner, I tried testing with a normal game object such as a sphere, to see if gravity was working properly. To my surprise, the ball fell to the ground and did not starting levitating to the sky. I then realised that the problem was with my game objects themselves and found that the problem lies in the animation, as I did not bake the animation’s rotation and position. So, the animation had its own rotation, which was separate from the game object, which caused the game object to bug out and fly upwards as its “gravity” was pulling it up.

Another issue I faced was when creating the object pool for my spawner. Initially, I used a list to hold all the game objects that were instantiated at the start of the game. I wanted to use a for loop to check through all the objects to see which was already active in the hierarchy and if it is, I will check the next object in the list, until there was one that was inactive, then I will use that game object to spawn and set active in the scene. So, the problem with this approach is that if I had 100 game objects in the pool, and I already have 99 objects in the scene which are active, I would have to loop through all 100 objects just to check if its in the scene. Now, what if all objects are already in use, the code will crash and stop working. This method does not allow me to use the objects in the pool efficiently and I was not able to reuse the objects, so I was not really implementing object pooling here. After doing more research and watching more videos, I learnt that I could use a queue instead of a list. Using a queue would allow me to queue up the next object to set active while placing the current object back to the end of the queue so that it can be reused. This allows me to avoid using the for loop and allows me to reuse the game objects efficiently. I was able to implement this into the particle system and it made the object pooling effective and working.

1. **Reflections**

**Reflect on:**

* **Your project, including the presentation and implementation parts. Discuss how useful you found the project work (e.g., relevance to game development, development of your technical knowledge, development of your programming skills, etc.) and how you approached it (e.g., your attitude, self-discipline, independence in learning, etc.).**
* **GMAPS as a whole. You can also use the discussion points suggested above here (e.g., relevance to game development, your attitude, etc.).**

1. **Project Reflection**

Your answer here (minimum 150 words for a *pass* grade) …

The presentation portion was interesting and fun as I was able to work well with my teammates and we were able to effectively share our research with each other. Since I was doing the case studies portion, I initially thought that it would be extremely boring as I had to find all the different games which used the particle system differently in their own way. But this turned out to be interesting as I was able to discover the different and unique ways, they implemented the particle system into their games. For example, God of War has different kinds of particle systems, one being for disintegrating the enemies’ bodies and one for the wind particles and wind direction in the game. Although both methods are using a particle system, their approach and implementation of it is very different from each other.

I also learnt many different techniques during this experience such as object pooling and passing variables to different scripts. Object pooling was interesting as it can be applicable to not just particle systems but example, in a shooting game, the bullets can be spawned with object pooling and this would minimize the workload on the CPU. Since, my implementation, I wanted the particle spawner script to be the one that is edited and changed on the editor, instead of having to go to every script and change one by one. To do this, I learnt about creating getters and setters for my variables which I would like to edit. I was able to implement this and create multiple variables in the particle spawner so my spawner would be reusable and customisable. Overall, I was extremely happy with this project experience as it taught me many things I did not know, especially about how diverse a particle system can get, and I am quite happy with my product as I felt that it was able to solve the problem statement I had created. I was also able to learn more about working as a group and learnt that communication is key when doing a project together.

1. **GMAPS reflection**

Your answer here (minimum 150 words for a *pass* grade) …

I enjoyed GMAPS as a subject as when I was in secondary school, I liked physics a lot and I also enjoyed solving math problems. GMAPS was able to teach me that game development has a lot of math involved behind the scenes, especially if I want to have a specific mechanic in game. For example, movement is a very crucial mechanic in most games and there is a lot of math and physics being involved. Making the character jump, would require me to calculate the gravity acting on the character and the velocity of the character so that I can calculate how fast and how high the character should jump.

One of my favourite topics from this class was vectors, as although it was a simple topic which was taught in secondary school, being able to use it in game development, for things like, look direction or calculating if an object is in the player’s point of view, was extremely fun and interesting to me. I never thought that this topic would be so relevant now, back then I thought that I would never need to use this topic and deemed it irrelevant in secondary school.

GMAPS taught me many different math and physics concepts such as dot product and homogenous coordinates, which are very important when developing a game. These are topics which took me some time to understand, so I cannot imagine trying to learn these myself without the help of the lecturer and the notes. Now I am feeling more confident in my own ability to make my own physics based game, and after the project, self-study and learning does not seem as daunting as before, so I hope to be able to make my own physics based game soon.