Post Mortem – Nikkolas Diehl 16945724

Overview:

I had quite a challenging experience with this project. I am relatively slow at programming and a bit of a perfectionist so often, throughout the project, I over analysed and took too long to work on what may have been small things.

But whilst I did make many mistakes, I am quite proud of what I did manager to finish in the tiny amount of time given.

I managed to complete my own hand made physics engine with accurate axis aligned bounding box collisions using impulse control and etc. I managed to complete a quadtree that hugely optimises the game and makes entity processing much faster. Especially in release build. I managed to get really nice sound design and sound system implemented and I feel like; overall, I did quite well.

What went right:

Out of the many things that went wrong or took too long, there was quite a lot that went right.

My physics engine is well thought out, inclusive and works with everything and every entity I could throw at it (mostly). All I had to do when creating new entities was insert them into the quadtree and the physics engine took the entities from it and made them automatically work. The physics engine probably took the longest (just over a week), but it went extremely right in the end and I am quite happy with it. In order to get this working, I first had to allow the physics calculations to happen multiple times per frame using an example of smaller timesteps. At the moment, I got it to be very accurate at 5-time steps per frame with a very smooth and solid execution. I then had to calculate and process velocity and accelerate based on forces and then apply friction to said forces to move my entities with gravity and input, and lastly, I had to detect and resolve collisions.

This definitely took the lonest time because I had to calculate the collision normal and find the impulse gathered from a given collision, plus having to calculate the collision intersection. It explains why this took so long.

My quadtree is also one of my proudest programming achievements. For about half a week, I worked on trying to get the quadtree with my custom axis aligned bounding boxes (AABB) to insert correctly and query correctly as well as remain optimized. In the end I managed to practically abuse smart pointers to make my quadtree run extremely well and smooth performance from over 4000 physics checks per frame, to under 200 physics checks per frame in extremely situations. The improvement was instant, immediate and extreme and I was extremely happy that it came to be so.

The quad tree takes in a single vector of all the given entities; alive or dead; within the scene and parses them into itself with an insert for each entity. If an entity fits with its width, height and x and y within a quad bound, it inserts that entity into that quad and then try and subdivides to fit more entities. This keeps going for every single entity and then when the physics engine queries a given range, all the quad tree has to do is return the entities found within that range. Immediately speeding up the physics of my game.

My animation system and UI system where also another thing that went extremely well and ended up being quite a fun system to work on. With how much work I put into it; when creating any animated entity, I could simply create a sprite sheet, specific a few values and dada. It worked. My UI system also fit very well into my animation system, as, for a button, I could feed it a sprite sheet with two different hover state panels and simply specify the details to the animation system and the system did the rest. Setting up my menu frameworks from then on became a breeze

What went wrong:

Whilst I am happy with what I did, there where many, many things that went badly or I had no time for.

Originally, I was intending to allow for a melee attach but due to time pressure, this did not make It into gameplay and thus there are a few issues and bugs with the AI when the run of ammo on all their guns.

The controller was another big issue for me and I ended up not having enough time to perfect it so right now, it is extremely buggy and doesn’t quite work. I never had time either to implement the ability to shoot with your controller so there is only basic movement and menu control.

I also never, as I originally intended, got a level system working and thus when you kill off all the enemies, there is no win state. There is, luckily a loose state when you die to the AI.

I also never got time to implement a blood particle system as I originally intended and due to this, I may get a lower grade than I wanted. There were many plans to spawn blood particles when an AI or the user died.

Lessons Learnt:

One of my main lessons was definitely to not try and perfect every little thing across my entire game as it wasted time and took away from the more important requirements to the game.

Another big lesson is to never underestimate the work load. I spent a long time procrastinating whilst working on big issues like my quadtree and instead wasted that time working on trivial expenses.

Conclusion:

In conclusion, I desperately wish we had more time to complete the task at hand as within the limited time we where given, I was not able to complete a lot of my original intentions. However, from what I did create, I do hope to carry forth my ideas, programming experiences and work that I have now done into future endeavours. If I’m ever creating another game sometime in the future, I can easily use my quadtree example as a great performance booster or my physics engine as a nice nostalgic tool.