Engine System Write-Up

[Download Engine System]

System Description

For the Engine System assignment, I implemented a networking system in my game engine that supports communication between multiple clients of a game. I made a UDP client-server model where data is passed to and from, and between the various clients, as well as the game server. Currently, the data being passed is the positions of the players themselves, and this is updated in real-time in accordance with the player's movement.

This system comprises two Visual Studio projects: the Networking project which lives in the Engine, and the GameServer project which lives in the Tools folder. The Networking project contains all the code related to the client side of the model, whereas the GameServer project implements the UDP server, which receives and sends data to update all the connected clients.

Both the client and server implementations contain three basic functions, to Initialize, Update, and Cleanup the respective data.

How To Use

- First, download the Networking and GameServer projects from the link above and add them to your solution. As mentioned earlier, the Networking project should be in your Engine folder, and the GameServer in your Tools folder.

- To run the GameServer, first build the solution. Then, locate the GameServer executable (it will be visible in the output folder of your temp directory). After you run the executable, you can start loading multiple instances of the game, and the GameServer window will show debug messages of positions it is receiving from the different clients.

- Additionally, you can also set the GameServer as a startup project in your Visual Studio solution. To do this, right-click the GameServer project in the Solution Explorer and select “Set as Startup Project.”

- In your MyGame project, you will have to create an instance of the client. You can do this in multiple ways. I declared a GameClient object as a member variable in the cMyGame.h file.

- In the cMyGame.cpp file, you will have to call the three functions related to the GameClient. In your Initialize() function, you will need to call something like gameClient.InitializeGameClient();

- Similarly, in UpdateSimulationBasedOnTime(), you should be calling gameClient.UpdateGameClient(); and in the MyGame CleanUp() function, you should call gameClient.CleanupGameClient();

That’s about it for setup and use. If you have any questions, feel free to contact me!

Findings and Learnings

First of all, I believe this engine system is far from perfect. There are definitely things I can improve in terms of usability and ease of implementation. However, the base features of a UDP networking system are all working well.

This was my first time diving into anything related to networking, so I was able to research and learn a lot about how multiplayer games (and networking in general) worked. I learned a lot about the two major data transmission protocols, TCP and UDP, including their advantages and disadvantages. Additionally, I also discovered how socket programming works in Windows in terms of sending and receiving data, which was a great learning experience.

From the previous assignments in this class, I definitely learned the importance of platform-independent interfaces and tried to implement the same code structure in my engine system. After I got the base implementation working, I refactored my code and made sure that all the Windows and socket implementations were in platform-specific files, with an independent public interface.