**Space Invaders TD Post-Mortem – Matthew VanCompernolle**

Space Invaders in essence is just a defense game, and my favorite type of defense games have always been tower defense games. For this reason, I decided to create a tower defense game that was loosely based on the classic game space invaders. The game is of course space themed and plays like a typical build your own maze tower defense, but also plays like space invaders in that you move a ship at the bottom of the screen and can shoot into the enemies in the maze. The main benefit of this design is that it makes the tower defense game much more interactive, and hopefully fun, than it usually is. One major goal I had with the project that had less to do with the gameplay and more to do with the implementation is that I wanted the engine to use a component entity system instead of a much more common class hierarchy system because I have never implemented one before. In the end I developed a fairly bar bones tower defense game using a component entity system that lacks many of the more interesting features that I ultimately desired.

The largest challenge I had with the game was definitely working out how to create an easy to use component entity system. I did a lot of research online before implementing it, but still ended up writing it three different ways before I was happy with it, and even then there are still some aspects I would change. Overall I am pleased with how it turned out, but I have found it takes me much longer to add certain features as I am not used to planning on how to make distinct chunks of data interact to make interesting mechanics. One benefit I was really hoping to achieve from the design was easy to implement multi-threading to increase game performance. It was easy to multi-thread, but the results are inconclusive. Almost all compiler settings make the game actually perform much worse on multiple threads, but in profiler mode the game runs flawlessly on multiple threads even with hundreds of objects on the screen. Why these odd differences in performance occurs still needs to be investigated.

Another challenge I had with the game was implementing a path finding algorithm for enemies so that they could navigate the maze be following a shortest path. I used Dijkstra’s algorithm and it worked well enough, but enemies moved only at multiples of 45 degree angles and it looked unnatural. What bothered me more was that it was clear that they were not following the shortest path. I came up with an algorithm to scan the clunky path that was calculated and remove waypoints that that were not used to avoid walls to smooth the path, ultimately making it the shortest path possible. There is still a bug however that I have not fixed yet where an enemy will go through a wall in very rare cases. This is most likely just a typo on my part, as a lot of the code was copy paste. I am overall very pleased with how the pathfinding turned out in the game.

For several reasons I was unable to add many of the features I want to have in the game, but I plan on continuing work on the project to add them in the future. I focused on the features that would make the game a basic playable tower defense and pushed most other features aside. Ambitious things such a level creator, a campaign mode, multiple spawn points, different enemy sprites and animations, an advanced upgrade tree, and more are still missing entirely from the game. Other things, such as some of the tower types and different guns for the player ship are also missing as well. None of these features are needed in the game, but they would make the game much more interesting. Without them, the game is a rather dull tower defense built on coding concepts I was curious to try. Depending on what the requirements of the next project are, I will either continue to implement the unique features I had originally envisioned in my current engine, or I will rewrite the game in Unity and place and much larger focus on the gameplay and features.