Ghosts

Maurice Leavell

1 Summary

The ghost of Epicman replicate the AI of the ghost of Pacman (except for the blue ghost whose AI is now the opposite of the pink ghost) in a 3d plane. Further information on the ghost AI can be found in the Pacman Dossier. The ghost are reliant on three classes: Ghost, Intersection Sensor, and Ghost Box. The Ghost box handles spawning and respawning of ghost and faced multiple problems in implementation, but the most jarring was calling spawn ghost custom event multiple times. The major difference of a custom event is that when it is called it runs the same exact instance and not a separate instances like a function, so when we called the custom event and it hit a delay it wouldn't make a new delay for this second ghost, it would completely stop the process. We couldn't use functions either to fix this because blueprint functions are not allowed to have delays so we instead had to make retriggerable delays that also stored if their was an excess ghost of the same type in the queue. This was definitely a class that would have benefited from being created in C++.

The Intersection Sensor was our way of not handling the game as an area of tiles with certain tiles having openings and others not. We placed box triggers using our grid class and triggers would destroy themselves if they spawn in a wall or if they only have a parallel path open to it. This was decided by for smaller triggers around the main box trigger that when spawned would overlap with walls and would deactivate that direction if there was a wall. Once the ghost stepped on the main box trigger the sensor called the ghost's change direction function and passed which directions were open for it to move in.

The ghost are all based on the same parent class that contains the change direction function. The only difference between the children ghost class is their ghost type enum and their color material that is used for the players to differentiate them. The algorithm for deciding direction is pretty simple. The target tile is retrieved for the particular ghost and whichever open direction is closest the ghost changes its direction to that. I looked into using Behavior Trees for this, but Unreal Engines behavior tree documentation was not well defined at the time I created this project so creating the AI in a blueprint function was a lot easier.