Abstract: Systems of a Racing Game

Josh Abramson

This semester in 3D Game Engines, we worked from scratch to build a functional engine including necessary game features such as an Entity-Component System, Object Collisions, Collision Optimizations through Spatial Acceleration Data Structures, Pathfinding, and Behavior Tree AI Systems. My capstone project took the engine I created over the semester and extended it, adding Procedural Terrain Generation, Linked Rigid Body Physics, and a Difficulty Scale, in order to create a high speed racing game for the player. This process involved creating a custom system capable of generating terrain at high speeds and recycling that terrain as the player moved in order to avoid memory leaks and crashes. It also needed to be able to be drawn and to run efficient collisions in order to function as part of a game, and not in isolation. Additionally, the physics additions required clever use of systems and certain helper algorithms to avoid rigid bodies with multiple pieces getting stuck. While this project did not serve a serious purpose, its implementation required me to consider how best to create these large, interlocking systems to allow them to function efficiently. While I could try to make greater pieces that encompassed more functionality, that created a tradeoff with flexibility that was not always helpful. It also required complicated math in the case of the collisions and physics, which made me consider how best to implement those for efficient calculations.