Project Report

Thomas’ Retribution

Module 3

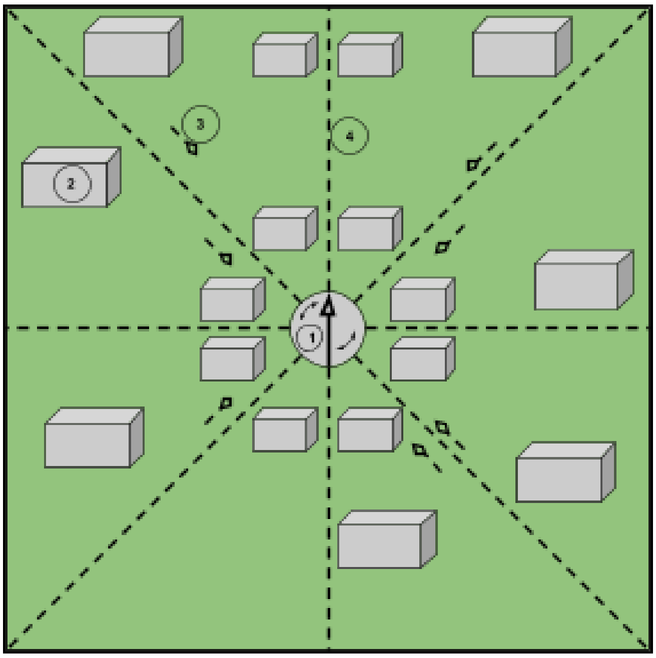
Game Concept

The player is Thomas the Tank Engine, who is now attempting to escape the train yard’s repair and service area. Enemies will approach Thomas from all directions and try to disable him for good. Thomas must use the coal he has available to annihilate his captors. Once enough are destroyed, Thomas will be able to claim his retribution and escape!

The Gaming Environment

Thomas’ final stand takes place at the centre of eight converging train rails. Thomas sits in the centre and has full 360 rotational movement on the X and Z axis as in the diagram to the right. The diagram is labelled as such

1. Thomas’ center turntable
2. Environment buildings
3. Enemies approaching the center
4. The railway lines connecting the center



Gameplay

Thomas must aim himself directly at the oncoming enemies, hold space to power up his “Coal Blast”™ and release to fire. The lump of coal will fire out of his chimney in a high arc into the air, making hitting Thomas’ enemies more difficult. As the coal collides with an enemy, the enemy is destroyed and Thomas is one step closer to his vengeance. Enemies appear in increasing numbers via waves. With the enemy count increasing by double per wave, until finally Thomas’ arch-nemesis Spencer spawns. Spencer is the final boss and will require a series of hits to defeat him. If Thomas is successful he is freed from the tyranny of the fat controller and his cronies and will chug free with his friends once more.

OpenGL Implementation

Gameplay

Waves of Enemies

There are 4 waves of enemies which increase in difficulty each iteration. Each wave the enemy count will double until every track has an enemy advancing down it. If the player survives past that point, a single Spencer train model will spawn down one of the tracks, be correctly rotated to aim the player and begin advancing slowly toward the player. Spencer will take 5 coal shots to be defeated.

Collision Detection

Collision detection is based on the position of the coal bullet being within a 2x2 radius on the X and Z axis and within 5 units of height on the Y axis. The position of the coal is tested by the center (0,0,0) point within the coal.

There is also a collision detection element that checks whether any of the advancing enemies have collided with a box around the player. Which if true will end the game.

Fog Implementation

Fog is implemented on a fragment shader level. It accepts the four dimensional vector Eye from the vertex shader, which is a combination of the view, model and position, in order to generate the fog distance and perspective correctly. In the fragment shader the density and color is set, and a small "getFogFactor" function is run to get a float which defines the density. This density is then multiplied with the fog Color and the texture to create the output of the fragment shader, color. The fog has an initial colour set to RGB(0.5, 0.5, 0.5) which is a dull grey colour. This will stay the same until spencer spawns and the skies will fade to a blood red colour RGB(0.75, 0, 0).

Bullet Drop

The coal's initial position is based on the camera's current facing. The coal's X and Z vector components are also based on the camera's current facing and that is the direction it will continue on. The coal's Y vector is initially set to 1.2. The bullet drop physics is as follows

- X = X * 0.992
- Z = Z * 0.992
- Y = Y – 0.015

The use of multiplication for X and Z slows the coal faster, whereas a flat decrease in the Y provides a smoother bullet drop curve.

Particle Effects

The particle effects are comprised of 2 components; the smoke from the bullet and a puff of fire when the bullet collides with an enemy. The smoke from the bullet creates a new 2D model of smoke every 5 frames that is initially rotated using the yaw of the camera so it is facing the players point of view. It is then given a position equal to the current coal position, however its initial position on the X and Z has a randomly generated float between -1 and 1 applied to it, this adds a level of variance to each of the smoke particles, so you aren't seeing the same smoke every shot. Each smoke puff is then translated upward on the Y and scaled down in size to give a fading away effect. For each of the smoke's iterations of upward translation, again a random float between -1 and 1 is applied to its X and Z again giving a more natural feel to the fading of the smoke.

The explosion effect takes place as soon as the coal collides with an enemy and is comprised of 3 fire sprites that have an initial upscale of 15 times their normal size. For each of the next 30-40 frames, each sprite will A. decrease in scale by 0.97 and B. translate upward on the Y by 0.1 until it has scaled to the point that it cannot be seen, when it is then reset and won't be attempted to render.