Computergraphics UE

Summary

We programmed a defend your base type shooter. We spawn at our camp and have to defend the base from the occurring monsters. Those are currently only walking to the pole and if they reach it the game is lost. If you shoot the monsters before they reach the pole by using the left mouse click you win the game.

Features in the game

Model Loading & Model Animation: We load a ninja warrior into our game and use it as enemies and animate it with gpu vertex skinning. Also we have shadow mapping

Standard shooter gameplay: We can move around using the WASD keys and using the mouse and SPACE to jump. Then we can shoot at our enemies with the left mouse button.

Light System: We use a point light and directional lighting.

Physics: Arguably the most time consuming task in our project was integrating the bullet physics engine into our game. We move our objects using forces and use gravity in our game. Collision detection is partly handled by the physics engine. For this to work we need to provide bounding boxes on our objects.

Multiple Level: We have 3 levels with different amounts of enemies and movement speeds.

Textured and Illuminated Objects

We are currently using point light and directional light for our scenes. The ground, start podium, crates and the projectile use the lighting information and are textured with the textures provided in the Assets folder. The crates can be jumped upon and the start podium can be made transparent.

The Trees are loaded and handled slightly different, but still lighted with the same lighting information and with the same shadow mapping as the rest of the scene. Because we had troubles with the physics on our camera object the avatar can run through enemies and trees.

Enemies use their own file format to load a bone structure and use the gpu for vertex skinning. They still use the same lighting information and the shadow mapping.

Libraries

Bullet: Physics library for collision detection and physics in the 3d world.

Assimp: Used in our Model Loading step.

FreeImage / SOIL: Used for Texture Loading. SOIL is used in the Model loading and FreeImage is used everywhere else.

Freetype: Used for adding text to the game.

Effects

Shadow Mapping: Every object in the Scene is used for shadow mapping.
http://learnopengl.com/#!Advanced-Lighting/Shadows/Shadow-Mapping

Cpu Particle Filter: Projectile use instancing to render 80 Particles for every projectile.
http://learnopengl.com/#!Advanced-OpenGL/Instancing

GPU Vertex Skinning: Enemy Objects use GPU Vertex Skinning for animating
http://ogldev.atspace.co.uk/www/tutorial38/tutorial38.html
http://www.3dgep.com/gpu-skinning-of-md5-models-in-opengl-and-cg/

Tools

Blender: Used for modelling.