Caroline Altrichter 01451760

Implementation

The architecture is broadly divided into the render state and game state in order to decouple the render logic from the game logic. Communication occurs via a message bus in an event/message based manner.

Engine.h contains both the RenderState and the GameState:
It creates the window, processes the input (all interactions implemented, intuitive controls needed for flying), wraps the basic game loop, loads the draw function and the IniReader file with the configuration settings.

Shader.h encapsulates the shader stages.
Pipeline.h encapsulates the whole shader program.
Light.h creates the directional light for scene lighting.

The **Geometry** class loads the geometry data (vertex data and index data) into VAOs and VBOs.

In **Model.h**, geometry which has been loaded using Assimp is associated with instances (model matrices).

Material.h and **Texture.h** provide material coefficients and other per material uniforms for the shaders.

Player.h implements the actual player (the Flying Boar plane).

BloomPipeline.h gives the hoops the green colored glow effect by performing post processing on an HDR render target and mapping it to an SDR render target.

GameState.h creates the actors as well as the hoops and checks whether the plane has gone through the hoop (sets and resets the scene).

Font.h is used for font rendering via character sprites.

Features of the Game

- 3D models are loaded with Assimp (the Flying Boar plane, the ground and the hoops)
- Models are loaded with various materials, e.g. texture of the plane
- Collision detection is provided by the Bullet physics library (if you want to see the collision use F2 to toggle rendering collision geometry)
- The plane is controlled by WS keys and the mouse
- The debug camera (F1) is controlled WASD, left shift and space (vertical movement) and the mouse
- The player/plane has to fly through the current green hoop, the next grey hoop will be activated when the green hoop was successfully flown through

- Game ends when all of the hoops have been flown through in the “correct” order. The Game keeps track of the player’s best time per session.
- To restart the game, press R.
- Toggling of the HUD (for the help menu) by clicking F3. HUD for the time that has elapsed and “best time” is always visible to the player.

Effects

(Points x1,5)

- Texturing: Specular Map (6 points) – shininess of the three stripes that are placed on the Flying Boar wings as well as on the logo which is also placed on the wings
- Advanced Modelling: CPU Particle System (12 points) – “fire” seen coming out of the turbines/jets.
- Post Processing: Bloom/Glow (12 points) – the green illumination around the hoops that the plane has to fly through. For the implementation I followed the tutorial from https://learnopengl.com/Advanced-Lighting/Bloom

Optional Gameplay

(Points x1,5)

- Collision Detection (6 points) – as seen in features of the game
- Heads Up Display (6 points) – used for displaying “best time” and current time passed (visible throughout the game), and you can toggle F3 to have a help menu displayed

Additional Libraries

Assimp: https://github.com/assimp/assimp/releases/tag/v5.2.3
FreeImage: https://freeimage.sourceforge.io
FreeText: https://freetype.org
GLEW: http://glew.sourceforge.net
GLFW: https://www.glfw.org/download.html
GLM: https://glm.g-truc.net/0.9.9/index.html
Bullet Physics engine: https://github.com/bulletphysics/bullet3/releases

Models

Flying Boar:

- https://free3d.com/3d-model/learjet-25-atlasjet-42833.html
  (The texture used for the jet I did myself)