

WINGMAN

DOCUMENTATION - SUBMISSION 1

Simon Pointner (01612401)

Marc Müller (01329181)

18.06.2018



Gameplay and Game goal

The game is a high score game. Therefore there is no win condition, only a loose condition. The player must fly as close as he can to the terrain to generate as many points as he can before he crashes with the terrain before he lands in the water.

Effects

Water Mesh

The water simulation is done by displacing the vertices and correcting the normals in the corresponding vertex shader.

Marching Cubes

The Marching Cubes algorithm is used for rendering the terrain, which is generated with the help of LibNoise.

Procedural Textures

The procedural textures are also generated with the help of LibNoise, by interpreting the noise values as color values. Slight adjustments had to be done to create the look of marble.

Features

Procedural generated Terrain

As already mentioned the game features a generated terrain in which the player must maneuver though. This is done with LibNoise and Marching Cubes in three steps: noise generation, vertex generation and normal smoothing.

Flight Simulation

The player movement simulates the movement of a wingsuit flight. This is mostly done by smoothing out the movement.

Text Rendering

Loading and HUD information is displayed with the help of FreeType, which generates Bitmaps for each character at the beginning of the game.

Controls

Controls have been implemented via `keyboard_callback` and `mouse_callback` functions.

Key	Effect
W, A, S, D	Move the character
Mouse	Move the camera
ESC	Close game
F2	Toggle fps display
F3	Toggle Wireframe Mode
F4	Toggle HUD visibility
F8	Toggle frustum culling

Illumination

Water and Player Model

The water is illuminated by ambient, diffuse and specular light.

World

The world is illuminated by ambient and diffuse lighting.

Text

The text is rendered with the help of blending.

Modelling

Blender was used for modelling the water and adjusting the character model. Also, the hierarchy for the hierarchical animation was made with Blender.

Sources

Additional libraries

GLM (mathematics library)

→ <http://glm.g-truc.net/0.9.8/index.html>

GLFW (OpenGL library)

→ <http://www.glfw.org/>

GLEW (extension loading library)

→ <http://glew.sourceforge.net/>

STB_IMAGE (image loader)

→ <http://nothings.org/stb>

ASSIMP (model loader)

→ <http://assimp.sourceforge.net/>

INIREADER (*.ini reader)

→ <https://github.com/benhoyt/inih>

NVIDIA PHYSX (physics library)

→ <https://developer.nvidia.com/physx-sdk>

GLAD (OpenGL loader)

→ <https://github.com/Dav1dde/glad>

Libnoise (noise library)

→ <http://libnoise.sourceforge.net/docs/index.html>

FreeType (text rendering library)

→ <https://www.freetype.org/>

Marching Cubes (scalar field polygonising)

→ <http://paulbourke.net/geometry/polygonise/>

Skybox

The Skybox was created with the Spacescape tool:

→ <http://alexcpeterson.com/spacescape/>

Models

Base for the player model:

→ <https://www.turbosquid.com/3d-models/minecraft-character-3d-model-1295104>

Textures

Water:

→ <http://seamless-pixels.blogspot.co.at/2012/10/tileable-water-texture.html>

Wood:

→ https://www.google.at/search?q=wood+texture+seamless&safe=strict&source=lnms&tbm=isch&sa=X&ved=0ahUKEwjulYykvd3bAhVRCuwKHe0yBB0Q_AUICigB&biw=1280&bih=931#imgsrc=j31XzLwZKfIMUM: