

Ida's Dream

Real-Time Rendering, WS18

Laura Luidolt, 1427250, Group 18

January 18, 2019

1 Implementation

The rendering is done with OpenGL 4.6.

The lighting is done using a simple Phong **illumination** model and only a single directional light. It is intentional that there are no shadows of and on the corpus of the scene (only Ida has a shadow), to allow easier 3D geometric illusions.

Textures are included on multiple objects (specifically the moveable ones) in the scene and Ida's crown and were made by myself using Photoshop.

The **scene**, see Figure 2, was modelled by myself using Blender and includes some special (hidden) features to enable 3D illusions. The **automatic camera movement** is specifically designed to hide all “problematic” angles and only shows a nice 3D illusion.



Figure 1: The scene.

2 Effects

2.1 Vertex Skinning (and Rigging)

The vertex skinning and rigging has been done on the main character's legs, so that there is a walking animation.

The vertex skinning was implemented using the ogldev Tutorial *Skeletal Animation With Assimp*¹ and the rigging was done in Blender using the *Blender Tutorial: Basics of Character Rigging* by Sebastian Lague².

2.2 Particle System

The particle system can be seen when the figure is walking, it leaves a sparkly trail behind.

The implementation was based on the *Revision Course Slides: Compute Shaders and Particle Systems* of this year's RTR Repetitorium.

2.3 Blob shadow

The main character has a single blob shadow at its feet, that is solely generated in the fragment shader. It is only drawn if the distance of the fragment to the feet position is small enough (specifically smaller than 1 unit) and uses smoothstep to allow a fall off of the shadow color.



Figure 2: All effects.

¹<http://ogldev.atspace.co.uk/www/tutorial38/tutorial38.html>

²<https://www.youtube.com/watch?v=cGvalWG8HBU>

3 Features

I implemented some features and other cool stuff, mostly for easier developing.

3.1 Animation

The animations are defined in an json file, which can be changed without needing to recompile (or even restart) the application.

Furthermore, the animation speed (and offset) can be defined in the settings file.

3.2 Reload

The settings and animation file can be reloaded during runtime (using the F2 key). Window parameters cannot be changed but everything else can be reloaded.

Furthermore the shaders can be reloaded using the F5 key.

3.3 Twist

The twist on some of the poles is done in the vertex shader only.

3.4 Waving flag animation

Like the twist, the animation of the waving flag on top of the corpus is only done in the vertex shader.

3.5 Single draw call

Using multi draw calls (i.e. drawing multiple objects with one draw call) and only a single shader the whole scene is rendered with only a single draw call. (Which is really cool??)

For rendering the particles a second draw call is necessary because of different primitive types (specifically points).

3.6 Orthographic camera

In the last submission I got feedback to use an orthographic camera instead of a perspective one, because it would fit nicely with the scene.

Indeed it does, so I am using an orthographic camera throughout all of the demo.

4 Keys and controls

Quite a few key options were implemented for easier developing.

F1 Restart the animation

F2 Reload the settings and animation

F5 Reload the shaders

P Pause the animation

N Next frame (only useful when the animation is paused)

T Print current animation time

C Change between automatic and manual camera

X Print pose of the camera (used for camera animation)

5 Additional libraries

Assimp: Geometry is loaded using Assimp³.

GLM: For the maths GLM⁴ is used.

irrKlang: In order to play music the irrKlang sound library⁵ is used.

nlohmann json: The json reader by nlohmann⁶ is used for reading the animation from a json file.

stb image loader: The image loader by stb⁷ is used for loading images/textures from files.

6 Github

<https://github.com/jercypackson/ezg18-IdasDream>

³<http://www.assimp.org/>

⁴<https://glm.g-truc.net/>

⁵<https://www.ambiera.com/irrklang/>

⁶<https://github.com/nlohmann/json>

⁷<https://github.com/nothings/stb>