

Computergraphik SS20 Easter Collection

Linda Ahmed 01525001

Anting Xu 01633092



Documentation for the second submission

Implementation and features from Gameplay list

3D Geometry

For loading our complex 3D models we use Assimp. Our complex models are the main character (the rabbit), and the carrots.

Advanced Gameplay and Collision Detection (Basic PhysX)

We implemented PhysX. We used PhysX for the implementation of the collision detection – you are not able to walk through a stone. We also used it for the Objects. Our rabbit jumps on the ground and is able to jump over a scaled stone.

Win/Lose condition

We have implemented a timer. You lose the game if the timer is 0 and you didn't collect all carrots. You also lose the game if you collect too many obstacles. You win the game if you collected 7 carrots. (You won is displayed)

Intuitive Controls and Camera

The Controls were implemented with the aid of Keyboard_callback and mouse_callback functions.

Source from : <https://books.google.at/books?id=bvFsDwAAQBAJ&pg=PA97&lpg=PA97&dq=processkeyboard+c%2B%2B+movement&source=bl&ots=ioFaXSvASZ&sig=ACfU3U2KBjAcmNZBJxssYuYgtN3PJWcKQA&hl=en&sa=X&ved=2ahUKEwi4gLvxzrroAhVDSBUIHUvZALAQ6AEwAHoECAkQAQ#v=onepage&q=processkeyboard%20c%2B%2B%20movement&f=false>

Intuitive Camera

You can enable a free camera movement, if you set the free attribute in the settings to true. We chose to set it to false so the camera follows the main character. Therefore there's a fixed offset.



Keyboard Key	Function
W, A, S, D, Space	Movement of the character and the camera – forward, left, backward, right Set the rabbit on the ground while jumping
Mouse	Move the camera around the world (if free camera = true)
ESC	Close the window
F2	Toggle FPS
F3	Toggle wire frame mode
F4	Toggle HUD
F5	Toggle Cel Shading
F6	Toggle diffuse and specular Map
F8	Toggle Frustrum Culling
F10	Toggle Normal Map
1	De-/Aktivierung - Rendern Map
2	De-/Aktivierung - Rendern Rabbit
3	De-/Aktivierung - Rendern Normalmap
4	De-/Aktivierung - Rendern Particle System
5	De-/Aktivierung - Rendern Animation

Illumination Models

The only light source we've implemented is a directional light. This means that all objects are lightened with the same intensity. The light comes from the same



direction. Ambient light, diffuse light and the specular light are calculated. We return the final Color. We also implemented a cel shader by the „CellShader“. The lighting values are calculated for each pixel and then then quantized to a small number of discrete shades. In our case there are four discrete shades. The cell shading can be toggled by the Keyboard key F5. We also use light maps, diffuse and specular maps with another light source on an object (Cube and Plane)

Textures

We load the textures via the stb_image loader class. The image data will be loaded to the texture units on the GPU.

Moving Objects

The rabbit is our main Character which is jumping all over the ground. The particles, the wall and the stars are also moving.

Adjustable Parameters

You can change the parameters like for example „free camera“, the brightness of the game, full screen etc. in the settings.ini file. The INIReader Class which was provided from the CG-Team helped us to extract the needed information from the file.

Heads-Up Display

For the HUD we used the FreeType Library. The counter, lives, and items as well as the FPS are displayed on the screen. The HUD can be toggled with the Keyboard Key F4.

Effects

Lightmap Using Separate Textures

We baked the two Objects (Cube and Plane) in Blender. The cube troughs a shadow on the plane behind. We baked our lighting information of our scene into a separate texture and combined our diffuse texture.

CPU Particle System

We implemented the particles by using instancing to create small stars which are also



animated. You can see how they are flying in the air in front of the wall. (under the Animated Stars). They are many small stars which are instanced and drawn with a single `glDrawElementsInstances`.

Hierarchical Animation

The 3 Stars are animated. We first computed the angles of the Objects, computed the position, scale, rotation and the parent transformation. The two stars are rotating around the middle one.

Video Texture

The Video textures flies in the middle of the room. We loaded the single frames for the video as single Images on a Texture.

Bloom/Glow

We used the Bloom/Glow Effect on our animated stars. The color of the Glow gets an own texture and then we blurred the texture that I wanted to render. So we ensured that our animated stars are shiny/glowy.

Simple Normal Mapping

You can see this effect on our moving wall. We used a colored texture as well as a normal texture on our wall. A light is directly in front of our wall so you can see the effect better. We also thought rotating the wall would be a good idea so the effect is more clearly to see.

Cel Shading

To be on the safe side if the simple normal mapping doesn't work we have implemented the cell shader which can be toggled via F5.

Specular Map

We also implemented this because we thought this is the „Lightmap Using Separate Textures“ effect and then we understood that it's not the same effect so we thought leaving it implemented will not affect us negatively. Therefore we used a second light source which is only used for the cube and the plane. You can see the effect by a white highlight on the objects.