Documentation

Group/Game Name: Run Pingu Run!

Brief description of implementation:
“Run Pingu Run!” is a brief 3-level game. Pingu moves through the map in search of his igloo. He tries to avoid getting eaten by seals or bumping into snow piles and getting stunned. Collecting fish helps him stay alive longer. Each level has a different terrain and increases in complexity by generating more enemies, snow pile obstacles and fish.

Additional libraries:
ASSIMP for object loading: https://github.com/assimp/assimp
Bullet for collision detection: https://github.com/bulletphysics/bullet3
Audio effects (we looked at how audio was used): https://github.com/OnixMarble/Titan-Voyager-Custom-Game-Engine

Gameplay:

Mandatory:

- 3D Geometry: Our models (penguin, seals, fish, piles, igloo) were made with Blender. We imported the models with the ASSIMP library. You can view them in the folder ***assets/models***.
- Playable: Pingu can move in all 4 directions and jump. The game can be directly started via the .exe file in the ***bin*** folder of ***executable2-runpingurun***.
- Advanced Gameplay: The sizes of the seals, snow piles and fish are initialized with a random scale. The said objects appear at randomly calculated places within a specific range on the map on every level. The seals move in randomly calculated directions. When a seal eats a fish, its speed is reduced. Collecting the fish helps you not get eaten and lose the game. You can bump into snow piles that slow you down.
- Min 60 FPS and Framerate Independence: The game runs at 60 fps and is framerate independent.
- Win/Lose Condition: You win if you clear all 3 levels by reaching Pingu’s igloo on every level. You lose if a seal collides with you, and you haven’t collected any fish beforehand or if the time for clearing the level is up or if you step outside the terrain. For both cases you get a distinct sound.
- Intuitive controls: We use similar keys that are used in other games for movement. We have the WASD keys (W – forward, A – left, S – backward, D - right). If you press any of them continuously, you’ll move in the defined direction. Space is for jumping. C is for camera toggle (see “Intuitive Camera”).
- Intuitive Camera: The camera can be toggled. We have a camera that is fixed onto Pingu at a specific angle as a 3rd person perspective that follows him when he’s in motion. The other camera is a free moving camera which can orbit around Pingu when moving the mouse left and right, change the angle view when moving the mouse down and up and zoom in and out when using the mouse

scroller. When playing with this camera mode Pingu moves relative to the direction of the camera, which is controlled by the mouse (he moves wherever the camera is pointed at).

- Illumination model: The whole scene is illuminated by a directional light. The fish represent point lights.
- Textures: Pingu uses a texture, which can be found under **assets/models** (Penguin_body.png). The “You win!” and “You lose!” signs at the end of the game use textures as well. You can find them in the folder **assets/textures**. We use a cube map for environmental mapping. You can find the textures in the folder **assets/textures/cubemap**.
- Moving Objects: The seals move on their own in randomly calculated directions within the bounds of the terrain. When a seal is fed, it reduces its speed. Pingu moves and jumps by pressing WASD and Space respectively (see “Intuitive Controls”).
- Documentation: This document.
- Adjustable Parameters: Parameters such as screen resolution, fullscreen mode, refresh rate and brightness can be adjusted from the **settings.ini** file in the folder **assets**.

Optional:

- Collision Detection (Basic Physics): We use the Bullet library for collision detection between objects. All objects plus the terrain have a defined rigid body. Pingu, the seals and the terrain have simpler rigid bodies for faster performance. The rigid bodies can be visualized with the F3 key.
- Advanced Physics: All objects have rigid bodies. Pingu and the seals move along with their rigid bodies. All hits between objects are registered by Bullet and are handled according to the game rules.
- Heads-up Display: Information about the current game state (on which level the player is, how many fishes Pingu currently has, how much time the player has to finish the level, a hint for quitting the game when pressing “ESC”) is displayed on top of the screen. The HUD can be toggled when you press the H key. We found this tutorial helpful for our implementation: https://learnopengl.com/In-Practice/2D-Game/Render-text

Effects:

Advanced Modelling:

- CPU Particle System: When Pingu moves, a long particle trial appears behind him. If Pingu hits a snow pile, particles fly from the top of the pile. We found this tutorial helpful for our implementation: https://learnopengl.com/Advanced-OpenGL/Instancing

Terrain:

- Tessellation from Height Map: We use a height map and tessellation shaders to create and display a curvy terrain. The rigid body has a smaller number of triangles that approximate the result of the tessellation algorithm for performance purposes. Depending on the height, different colors are used in the fragment shader to mimic water and snowy surfaces. We found this tutorial helpful for our implementation: https://learnopengl.com/Guest-Articles/2021/Tessellation/Tessellation. Our heightmaps in assets/textures are smaller cut-outs taken from the tutorial from the previous chapter of the same website: https://learnopengl.com/Guest-Articles/2021/Tessellation/Height-map

Animation:

- Hierarchical Animation: Pingu moves his feet when in motion. He can move his eyes left to right. He can open and close his beak while simultaneously moving his eyebrows. The mentioned moving parts are separate meshes that rotate at an appropriate angle.

Texturing:

- Environment Map: For a more realistic feel of the game a cloudy background environment was implemented with cube maps. The clouds also reflect onto the whole terrain. We found this tutorial helpful for our implementation: https://learnopengl.com/Advanced-OpenGL/Cubemaps

Shading:

- Cel Shading: Pingu, the seals, the fish, the snow piles and the terrain have a segmentation of the gradient. This effect is implemented in shaders and can be switched on and off with the F4 key.

Post Processing:

- Contours via Edge Detection: Pingu, the seals, the snow piles and the fish have a distinct contouring. This effect is implemented in a post processing shader and can be switched on and off with the F5 key. We found this website with examples helpful for our implementation: https://www.shadertoy.com/results?query=sobel

For implementing post-processing effects such as Cel Shading and the Contours via Edge Detection we used this tutorial: https://learnopengl.com/Advanced-OpenGL/Framebuffers

Walk-through:

Objects:

- Pingu: You are Pingu (the penguin). You can move around with the keys WASD (forward, left, back, right) and Space (jump). Pingu appears at a random position in the near left region of the map.
- Seals: They are the seal models that appear at first at random positions on the map away from Pingu and move randomly. If you collide with one without having collected any fish beforehand, you lose and get a “You lost!” sign. If you have fish, you won’t get hurt and the seal will turn a darker cyan because you “fed” it and now you have one fish less. If the seal collides with a fish, it will again turn a darker cyan because it ate the fish and will slow down its speed. A seal that is fed (is a darker cyan color) is no longer a threat to you even if you bump into it without any fish.
- Fish: They are the pink figures that glow. They appear at random positions at lower levels of the terrain (like they are in water). They don’t move. If you collide with one, you

collect one fish, and it disappears. If a seal collides with one, it disappears. Upon collision you can hear a pop sound indicating that the fish has been collected.

- Piles: They are the white-grey rock-like structures. They don’t move. If you collide with one, you jump back and are stunned, which means you can’t move for a bit (2 seconds).
- Igloo: It appears randomly in the far-right corner of the map. If you collide with it, you start the next level or win and get a “You won!” sign if you are on the 3rd level (which is the last level).

How to play:
There are a total of 3 levels. Each new level increases Pingu’s speed, the number of seals and their speed, the number of snow piles and fish.
Your goal is the same for each level: Reach the igloo. Do so by moving towards it and avoiding collisions with the seals and snow piles. Collect fish to ensure you won’t get eaten. If you clear all 3 levels, you win. If you walk outside of the terrain or get eaten, you lose.