TSBK07 — Project report

Erik Håkansson Kjellman – eriha181 Klara Wingstedt – klawi021

May 20, 2015

1 Introduction

The specification for the project was as follows:

–Will do:

- Lunar lander 3D model with textures
- First person camera
- Terrain
- Light
- Skybox
- Goal position
- Water

–Might do:

- Particle system (fire, explosions?)
- Auto-generated terrain
- Trees
- Sun(s)
- Score
- Highscore list
- Welcome screen
- Sound
- Auto-generated textures (filtered perlin noise etc)
- Third person camera (fixed on terrain)
- Craters
- Fishes!
- Under 64K

All of the ”will do” items were implemented, as well as a few from the ”might do” list, namely auto-generated terrain, welcome screen, auto-generated textures and third person camera.

2 Background information

We wanted to make most of the elements of the game auto-generated or a slight bit random, partly to make the game more fun and natural, but also as a challenge.

3 Implementation

The project was implemented in C with some helpful code given from the lab series, mainly VectorUtils3 and loadobj. The spaceship was modeled in 3D Studio Max, the text models were made using Inkscape and OpenSCAD, and most of the textures were made in Inkscape.

Most of visual objects in the game are described as structs in the code, although the fuel bar, welcome screen, game over or win screen are bundled together in a struct called HUD as in heads-up display. The game loop is run 50 times a second, and updates the graphics and all movement code, as well as uploading variables to the shaders.

4 Problems

Each cloud in the game is made up by 10 spheres which are randomly placed close to each other, they are also randomly scaled in all axes. This proved to be very nice looking but also very GPU intensive. This could have been solved by using LOD and/or billboards further away. The terrain was generated with perlin noise, which came out nicely, however it took a while to get it working and scaling it properly for our purposes. Shader programming was new to us before this course, so that paradigm took a while to get used to. After a while it became very intuitive and easy to work with, especially with how quickly you could get nice looking results.

5 Conclusions

We are quite happy with the result, our main wish would be to implement a real spherical moon that you can actually circumnavigate. We started on an implementation like this, but were unhappy with the result of that.

6 Images

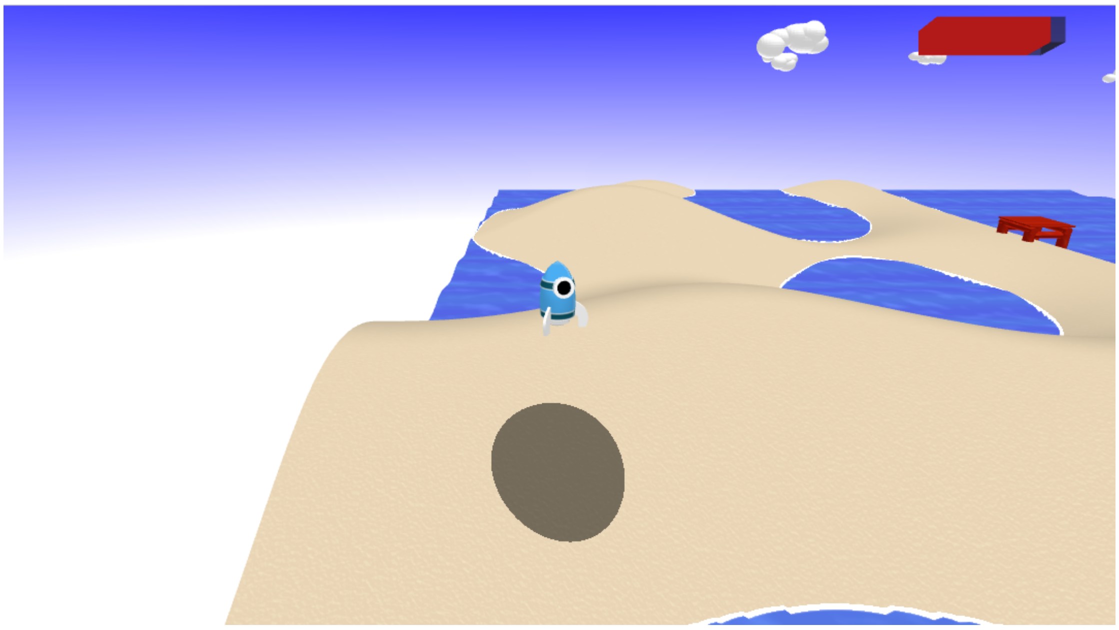

Figure 1: This is what the game looks like when you start it.

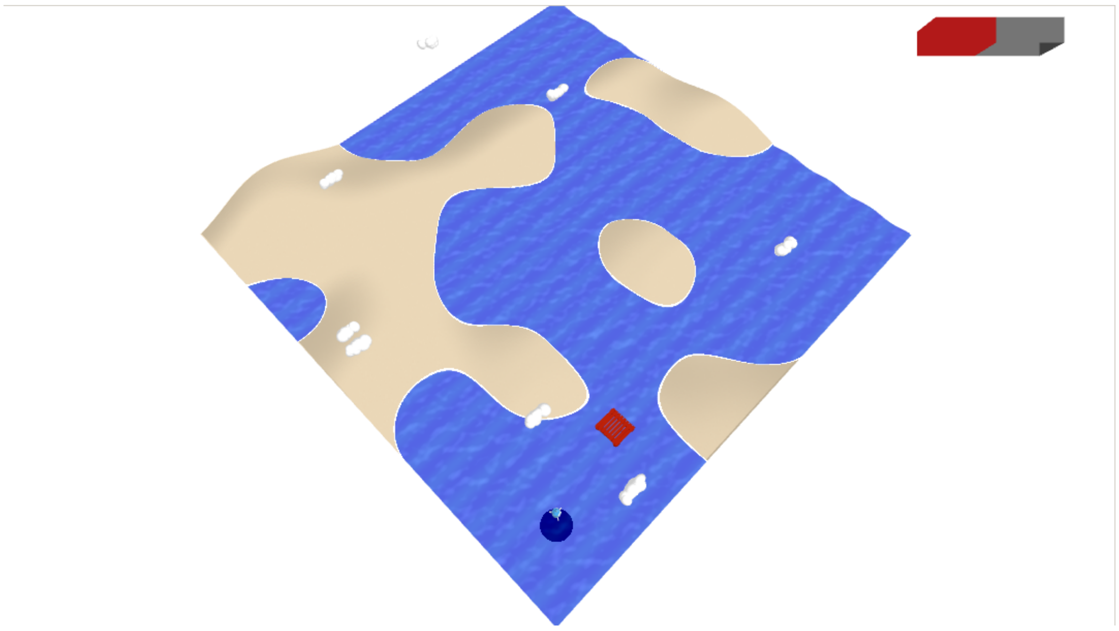

Figure 2: This is an overview which you get by holding down 'c', used for finding the target.

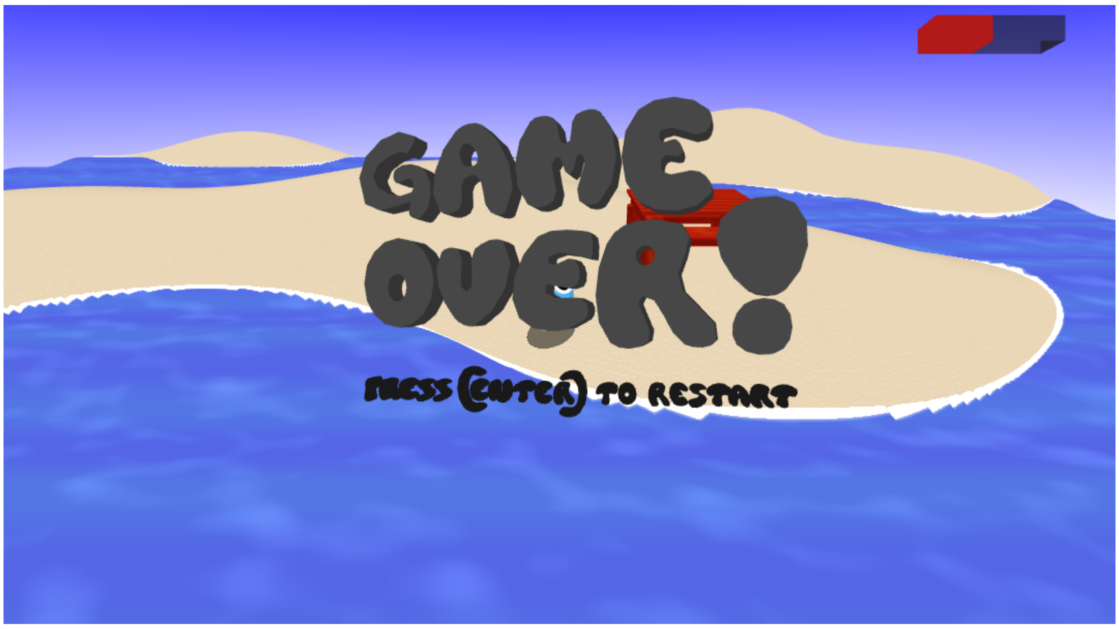

Figure 3: Game over screen.

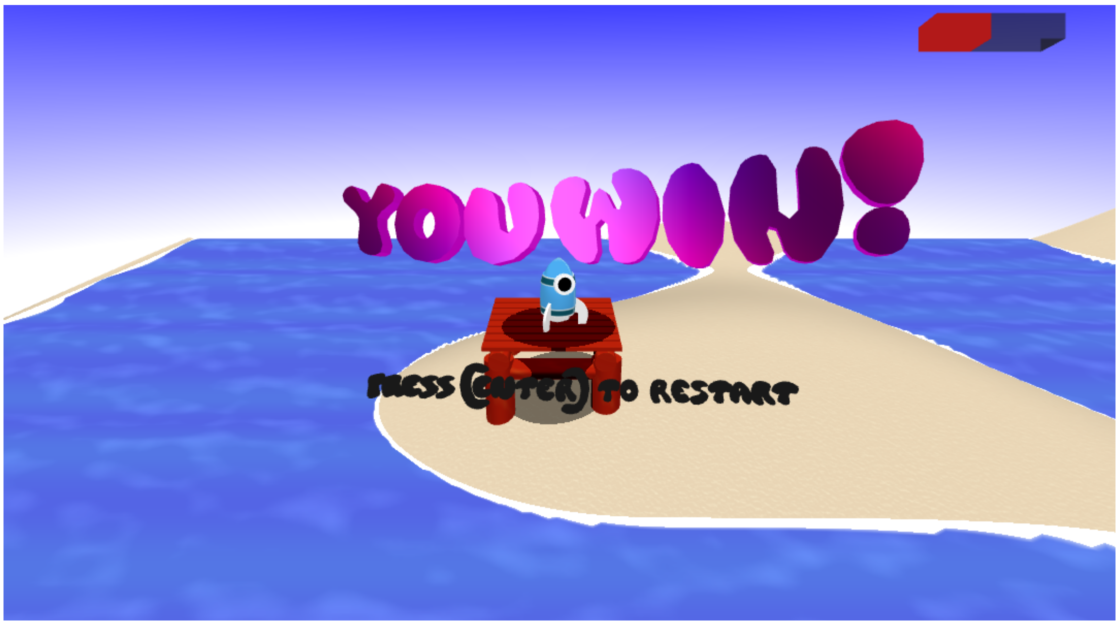

Figure 4: Winning screen.