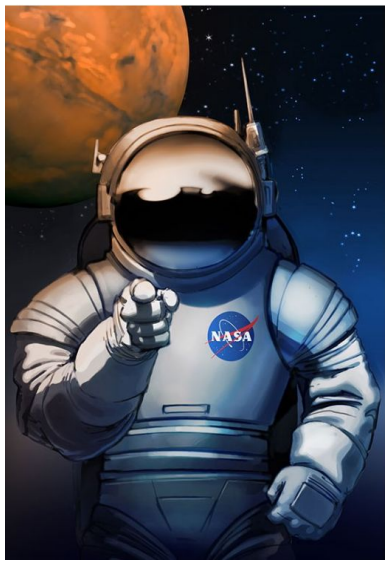
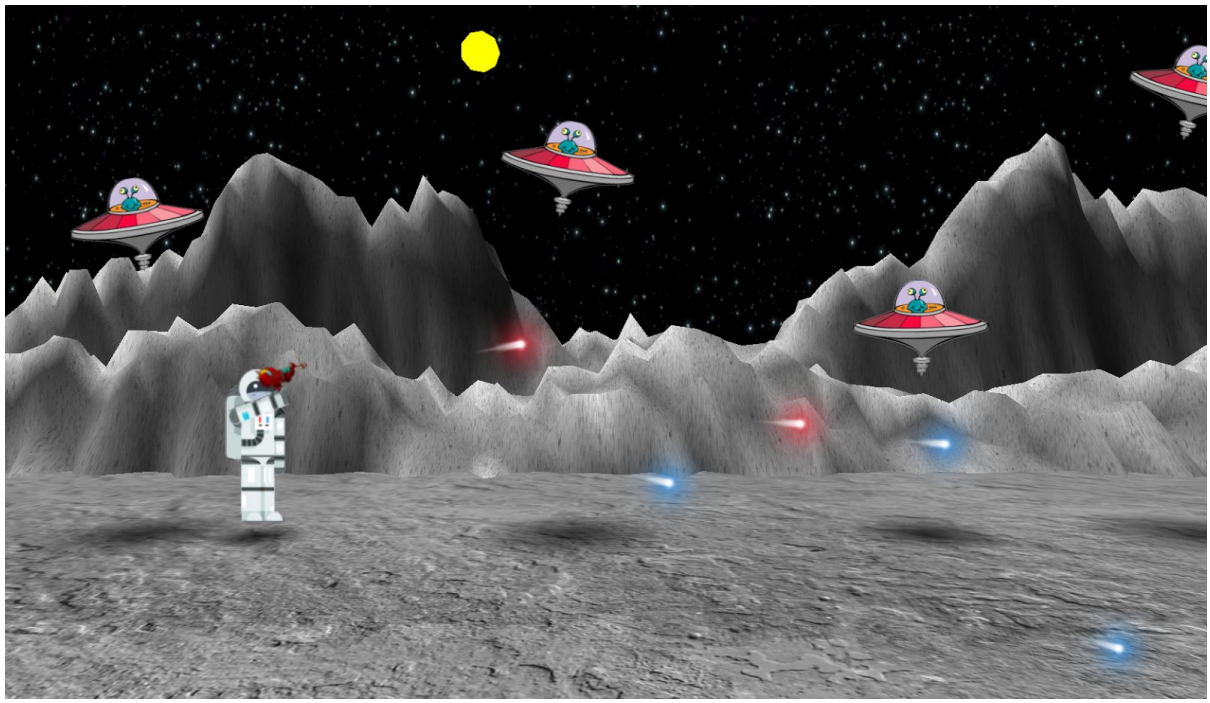


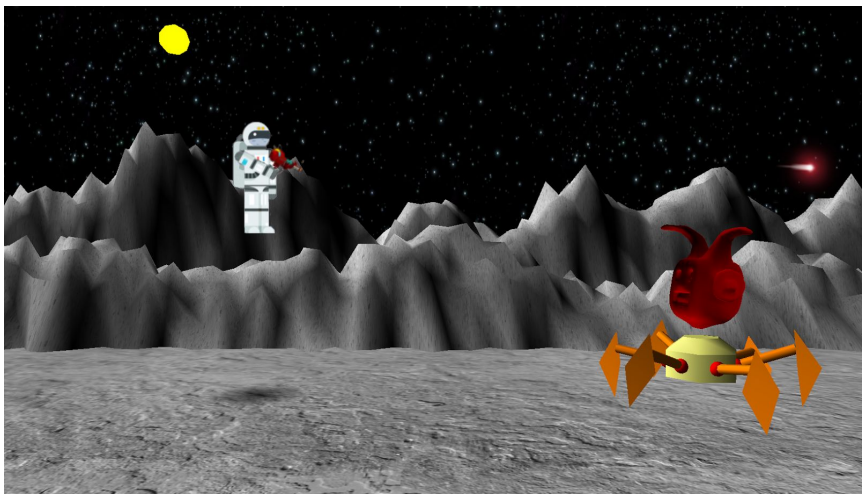
Report on
“An Astronaut’s Mission”

IN4152 - Computer Graphics and Animation
16-06-2016

Matthijs Amesz - 4010620
Dustin Lim - 1514202
Wouter Groen - 4001362
Marlou Pors - 4008847

In this small report we will shortly discuss the components of our game “An Astronaut’s Mission”, which we build for the course IN4152.





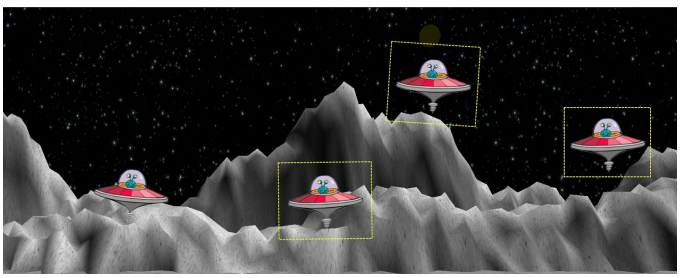
Character & Enemy Building

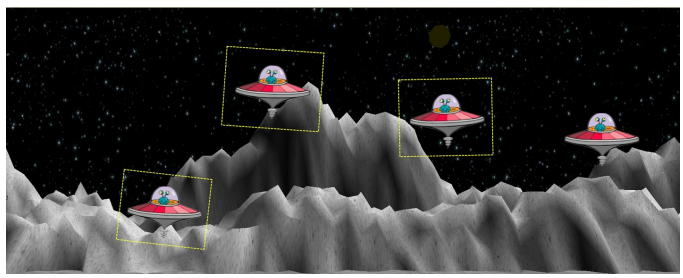
see files: “Entity”, “main”

For this we use 2D textures. The character uses 3: one for the body, the arm and the gun.
The arm (and body orientation) follows the mouse. Although the arm rotates at the shoulder, the reference point used to calculate the angle towards the mouse is the yellow sphere (see image) - located at the height of the gun-barrel. This point is also used to get the correct spawn point for the bullets.

The bullets flicker between blue and red.

As an extra, the enemies fly in a sinusoidal manner, while they also wobble, made clear using the yellow boxes below.

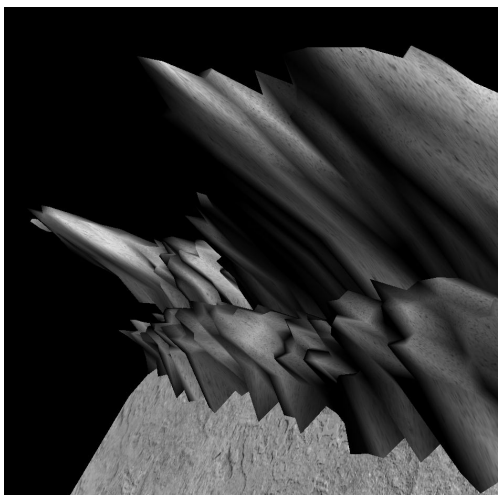


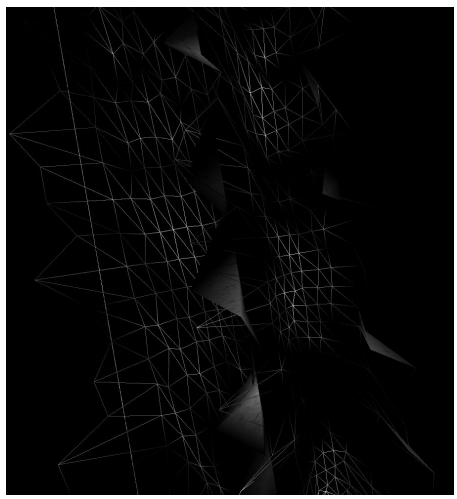


Landscape Creation

see files: “landscape” (“background” for the ground and starry background)

We follow exercise 4 with a complicated sum of sinusoids. Then we add a random height to achieve the end result (left). The Z-value determines the height to create 3D ridges.
Right shows how the mountains are constructed using the vertices.

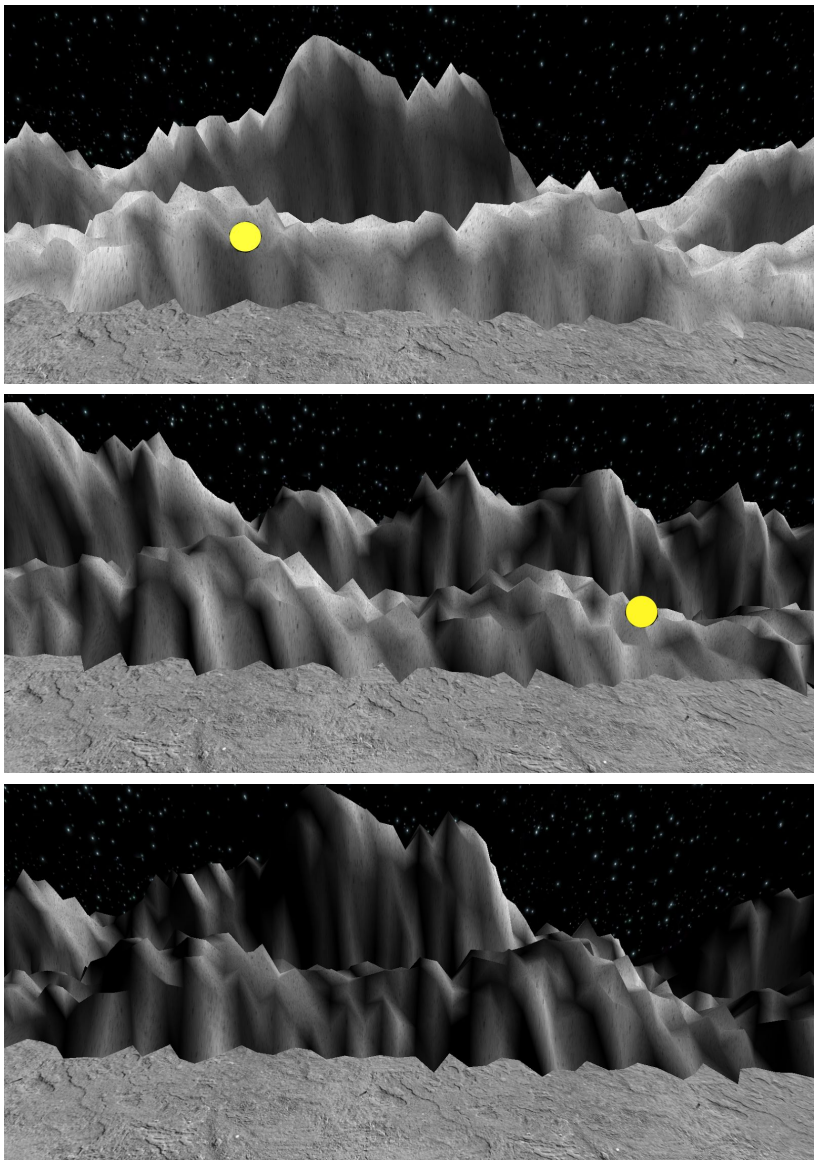




Landscape Lighting

see files: “main”

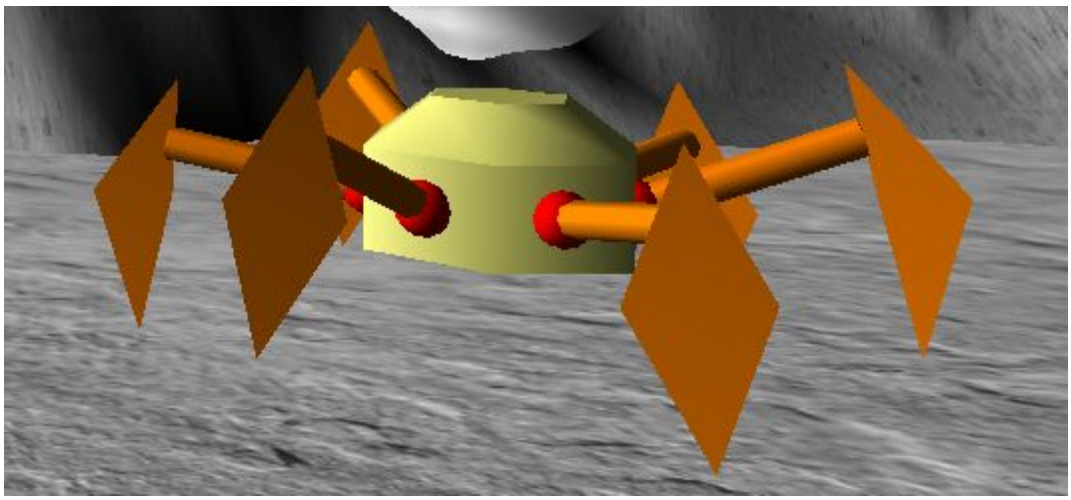
The game uses a distant sun as light source, which moves from left-to-right above the world. These three images show diffuse lighting with the sun positioned left, top and right respectively.



The Bossfight

see files: “boss”

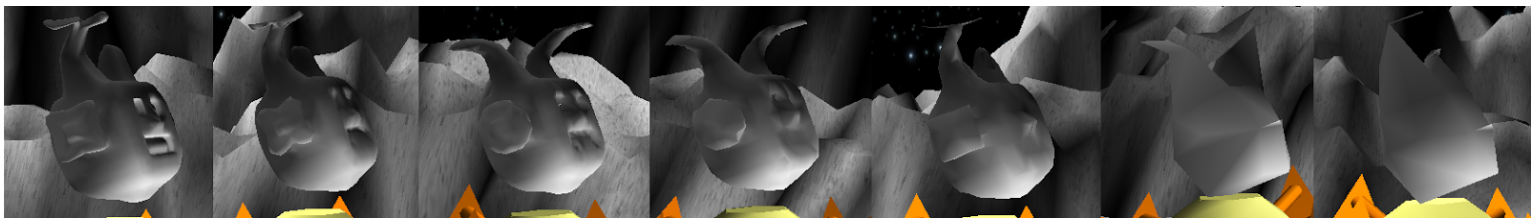
The boss is created out of 2 main parts: it’s body and it’s head. The base of the boss is a sectopod, a walker with 6 legs. The legs have a realistic walking animation, which is completely calculated by the code.



Mesh reduction

see files: “boss”, “hoofd”, “grid”

The boss’ head is created in Blender, and goes through several stages of mesh reduction when hit by the player.



Phong Shader

see files: “main”, “mesh”

We implemented a Phong shader, which calculated the light intensity per vertex. The intensity information was multiplied with a colored light and applied on a white mesh to produce the result. The lightsource is always positioned below the head, but the head itself rotates so you can still observe the dynamic shading effects.