Sapienza – University of Rome

MSc in Engineering in Computer Science

Course of Interactive Graphics

Star Wars video game developed in

Three.js and WebGL

0. Summary

1. Environments

a. Frameworks

As environments, it has been used three.js and WebGL.

Three.js is a cross-browser JavaScript library used to create and display animated 3D computer graphics in a web browser.

Three.js uses WebGL.

WebGL (Web Graphics Library) is a JavaScript API for rendering interactive graphics within any compatible web browser without the use of plug-ins.

b. Tools

- Brackets coding

  Brackets is a modern text editor that makes it easy to design in the browser
- Dropbox file organizations

  Dropbox is a modern workspace designed to reduce busywork-so you can focus on the things that matter
- Microsoft Office documentation

  Office is a suite of tools, including word processor, spreadsheet and a presentation
- Skype calls

  Skype is a software that provides video chat and voice calls
- GitHub version control

  GitHub brings together the world's largest community of developers to discover, share, and build better software
- LinkedIn and SlideShare sharing

  LinkedIn and SlideShare let you manage your professional identity

2. Models

a. Imported models

From clara.io we imported a 3D JSON model of the X-Wing, a famous spaceship of the Star Wars universe.

Figure 1: X-Wing

b. Simple models

We created a simple three.js model to represent rocks, which is a tetrahedron.

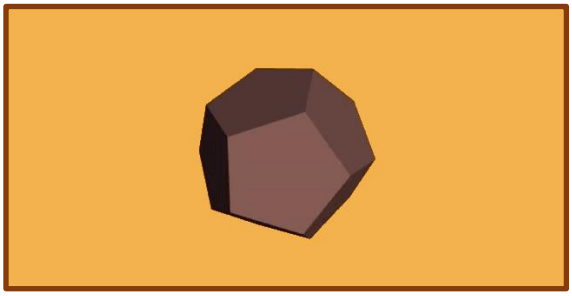

Figure 2: a rock made with three.js

c. Hierarchical models

BB-8 droid: a sphere, a semi-sphere and four cylinders.

Figure 3: BB-8 droid hierarchical model without textures on it

3. Shadows and lights

a. Shadows

PCFSoftShadowMap filters shadow maps using the Percentage-Closer Soft Shadows (PCSS) algorithm using renderer.shadowMap.type = THREE.PCFSoftShadowMap.

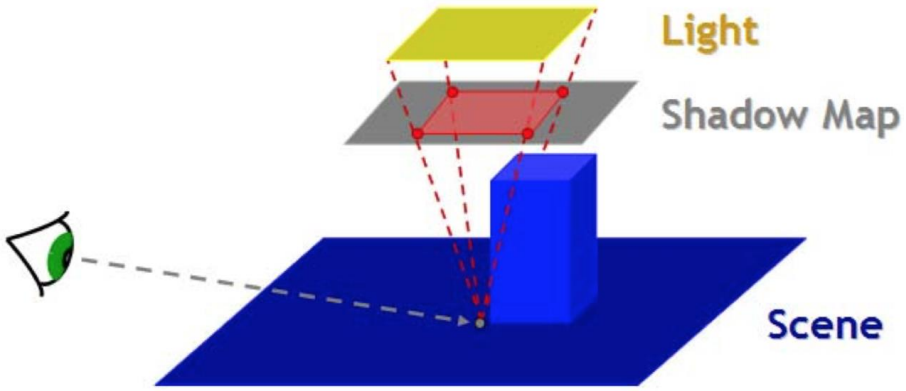

Figure 4: light and shadow map on the scene

b. Directional light

It is a light that gets emitted in a specific direction. This light will behave as though it is infinitely far away, and the rays produced from it are all parallel.

The common use case for this is to simulate daylight; the sun is far enough away that its position can be infinite, and all light rays coming from it are parallel.

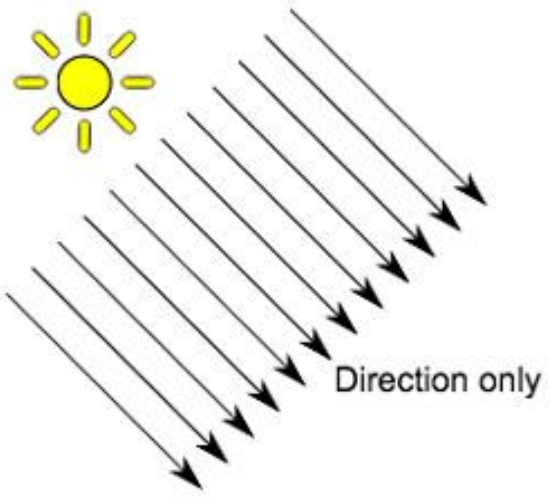

Figure 5: directional light example. The Sun is so far that its light rays are parallel

c. Lambert material

For the BB-8 model, it has been used the Lambert Material: shading is calculated using a Gouraud shading model. This calculates shading per vertex and interpolates the results over the polygon's faces.

It allows to obtain a soft degradation from light points to shadow points.

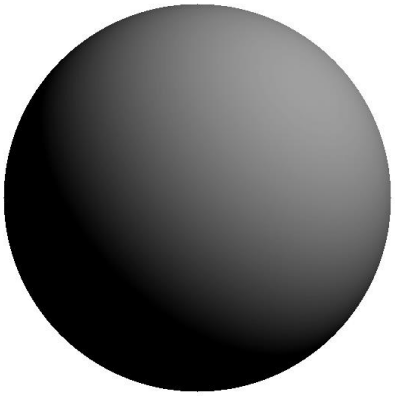

Figure 6: soft degradation from light points to shadow points

4. Textures

In the visual arts, texture is the perceived surface quality of a work of art. It is an element of two-dimensional and three-dimensional designs and is distinguished by its perceived visual and physical properties.

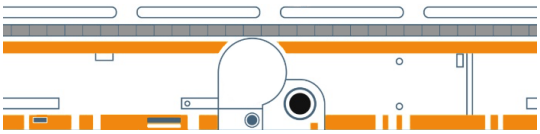

Figure 7: head texture

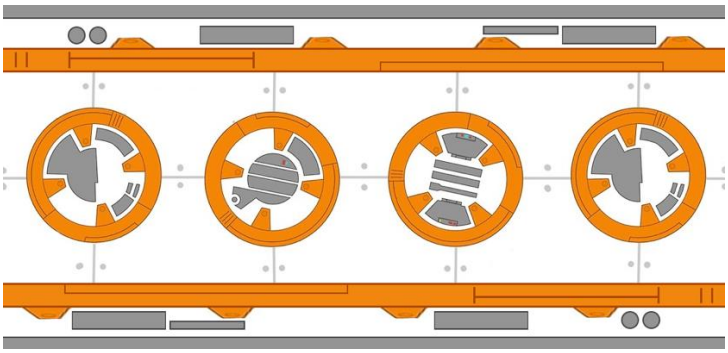

Figure 8: body texture

5. Animations

Animation is a dynamic standard in which images or objects are manipulated to appear as moving images. Three.js allows the creation of Graphical Processing Unit (GPU)-accelerated 3D animations using the JavaScript language as part of a website without relying on proprietary browser plugins.

Figure 9: BB-8 droid with textures, running and jumping

6. Business logic

a. Start Screen

The start screen is a star field with the model of the X-Wing flying in it. The stars are generated casually through a loop where the coordinates x and y are assigned randomly, while the z position matches the iteration variable of the loop. Once the button "Start" is clicked, the Animation Frame of the start screen is deleted and the game scene is drawn, where a new Animation Frame is created.

b. Game's world

The entire world game is a whole and enormous sphere, which rotating gives to the observer the sensation of moving on it. The sphere is generated with `THREE.SphereGeometry`, it is not a real sphere but a geometric solid composed using many triangles and characterized by some parameters: ray, longitudinal segments, latitudinal segments, phi and theta.

It has been decided not to use many triangles to make the world not perfectly rounded and spherical, giving the sensation of a non-perfect and more natural surface. Also, because we are zooming on it, it should be very difficult to make such a big world perfectly spherical, while BB-8 was easier because of its smallness and distance from camera.

Phi and theta are at their maximum values because the world sphere need to be a complete world, while these parameters lets a sphere to be not completed (such as a hemisphere).

Instead of a monochromatic background that gives the sensation of a flat and less natural world, it has been applied as background an image with the same style of the world sphere, giving the sensation to reach it and it lets the users to see a beautiful panorama (remembering also that Jakku has two Suns!).

c. World's population

World is populated with rocks. Rocks, as anticipated before, are three.js models which represent a tetrahedron. They have a collision logic that lets them to be attached to the world's surface and let the game ends when the player, moving BB-8, collides on a rock.

Another logic that characterize rocks is the spawn: rocks casually spawn with a random position on the world's surface and with a random dimension, making the game more difficult and less predictable.

d. Game's player

The game has a player, which is the user interacting with BB-8 droid. BB-8 droid is a hierarchical model (as described before) and it stands on the same y axis even if it seems to move forward. BB-8 can move only on the z axis jumping and falling down simulating world's gravity and it can move on the x axis, trying to avoid rocks and the consequential game over.

7. User interaction and manual

a. How to use this manual

This software is a challenging video game, addressed to those who want to test their personal skills. Users can use this easy-to-read and portable manual wherever they want, at home, in the office, at a remote job site or even in the car (if you're not driving!). Users can start reading this manual from "Getting started" section skipping this introduction. This manual also doesn't need expertise or knowledge to be used, thanks to its simple language and its full-provided steps and instructions, also, it doesn't include high technical language to help those who don't have technical background.

b. User interactions

User interaction researches the design and use of computer technology, focused on the interfaces between people (users) and computers.

In our video game, the user can take the control of BB-8 droid and move it using keyboard's arrows; it will be discussed in the next chapter.

Figure 10: a user interacting with computer

c. Getting started

To start the video game, users can either navigate to the GitHub page () or download the repository. If it's chosen the second method, users have to open a terminal in the folder containing all the project and type:

```
$ python -m http.server
```

To start a simple http server using Python 3.x, or:

```
$ python -m SimpleHTTPServer
```

For Python 2.x. Now just go to "localhost:8000" using any browser and the game will start.

Note: you should clear your browser's caches sometimes, or your browser could not be supported. If you found any trouble, try to clear your caches and to use Google Chrome or Safari.

d. Play the game

At this point the software should be running and you should be ready to play it.

In the browser, just click on START button and here you go! The video game will start.

You will impersonate BB-8 droid trying to escape from the imperial army, running for your life while avoiding rocks jumping (by pressing ↑ arrow) and moving left (← arrow) or moving right (→ arrow).