SURVIVE ON MARS

INTERACTIVE GRAPHICS FINAL PROJECT

CELLI MICHAEL 1968299

THE GAME

Survive on Mars in an endless runner game; it is about a WebGL based game, for which I've used the JavaScript programming language to build the logic of the game, with the ThreeJS library, which simplifies hugely the writing of webGL based applications, since it provides a layer of abstraction over the bare webGL implementation and TweenJS to achieve the animations. Obviously, being a web-based game, it also requires some html, with some css for styling. For all the libraries I adopted the CDN way, so all the used libraries are not included in the project but are downloaded when starts.

As the name suggest and as is showed in the Figure 1, the game is set on mars. An astronaut runs on the mars surface trying to avoid meteorite pieces. With the astronaut there is a small rover which is there to scan the mars terrain with the camera and follows the astronaut during the running. The objective of the game is to survive as long as possible, avoiding these pieces, making the best score with the longest run. The speed of the game increases while the astronaut advances thought his run. Every time that the astronaut collides with a meteorite, he loses 10 health points and when his health is 0 the game will end. The user can also, at the beginning, set a difficulty (easy, medium, hard) and this will affect, increasing it, not only a fastest rotation of mars surface but also a shorter respawn time of the meteorites. It's important to know that the meteorite parts are respawned after a few seconds (depending on the difficulty selected) that the last interaction with the astronaut was done. So the criteria is that when few seconds are passed after the last interaction, for sure there is nothing in front of the astronaut so meteorites must be showed. Every time that new meteorites are spawned, the orientation of them might be changed from the previous tranche. Orbit Controls are activated to look around and inspect the 3d world created.

Figure 1: Gameplay

3D OBJECTS AND ANIMATIONS

In the game, I used five 3d models downloaded from Sketchfab (mars, astronaut, rover, meteorite) and to complete the space environment I created other two models directly with ThreeJS (earth, mercury) and applied to them proper textures. In the pictures below are represented two of the 3d models used; in particular, we can see the starting position of the astronaut in the Figure 2 and the rover in the Figure 3.

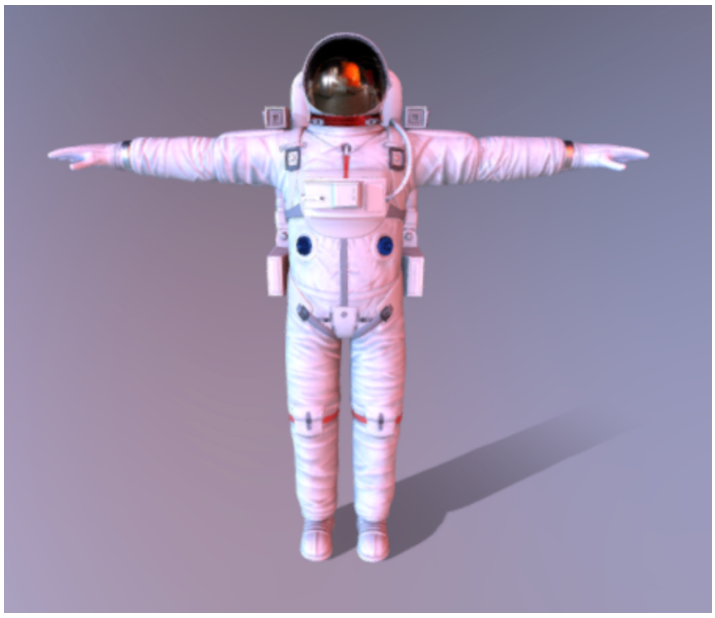

Figure 2: The Astronaut

Figure 3: Explore T30 concept rover

Both the astronaut and the rover are hierarchical models made of many different parts (bones and meshes) that can be manipulated from the javascript code with ThreeJS; of course, the astronaut is a long way more complex and detailed than the rover. In fact, every single part of the astronaut can be manipulated independently to achieve whatever animation you want. To complete the desired animations, I used at most 18 parts together. For the rover, the work was a lot much simple since not all the single parts can be controlled alone and also because the animations were simpler.

For the animations, all were created using TweenJs which is a Javascript library used to create smooth animations. In the game I developed 4 different animations for the astronaut and 5 for the rover. In particular:

- Astronaut running
    - To start the running animation, first I setted the astronaut in the running position (with one leg in front of another, arms curled ecc) performing some rotations on all the three axes. Then I create a tween animation, in loop with the .yoyo function to simulate the running on the mars surface.
- Astronaut jump
    - Since the astronaut must avoid the meteorite pieces, he can jump over them and continue on his way.
- Astronaut sliding left/right
    - Same concept of the jump, but now the astronaut slides on the left side
- Rover sliding left/right (following the astronaut)
    - The rover follows the astronaut, sliding left and right
- Rover wheels rotation (both for left and right)
    - When the rover slides left and right, I created also the animation for the wheels in such a way that they are directed left or right depending on the sliding direction
- Rover scan camera
    - There is also a continuous animation on the rover, in particular the topping camera rotates 360 degrees to simulate the scanning of the surface

All the imported models are .GLTF, so in the project the GLTF loader has been used. The extension .GLTF is the one recommended to be used inside ThreeJS projects.

LIGHT AND TEXTURES

In the project I used two different lights:

- Ambient light
- Direction light
    - This simulates the sun light on the planet, in fact on the mars surface objects shadows are showed

For the textures, all the 3d models imported had their own textures. I used a bump map with a color texture for the mercury planet and a color with a normal map for the earth.

USER INTERACTION

The command to play the game are very simple

- Left/right arrow to move the astronaut left/right
- Space bar to jump

As said, at the beginning the user can select the difficulty of the game. Inside the scene, the user can mute the audio added (background music, collider sound and gameOver music) through a simple button.

BROWSER PERFORMANCES

The game has been tested for the most common browsers (firefox, google chrome, safari). The performances on google chrome are not high as the ones in firefox or safari. For this reason, I suggest to run it with firefox.