Three-Dimensional Vizualization and Animation
Technical Report - Assignment III.

Marfeychuk, Mykhaylo
ist194039

Skalický, Matyáš
ist194904

1 Cameras and stereo effect

The application contains several cameras. The player can switch the camera using the keys 1 to 5. All of the cameras also show information about the game except for the stereo mode, for which we do not show any information on the screen to make the screen less cluttered.

Figure 1: From top left: perspective following camera, rotating perspective camera, top perspective camera and top orthogonal camera.

The stereo viewing was implemented based on the module provided on the ThreeJS demo accessible at `https://threejs.org/examples/webgl_effects_stereo.html`.

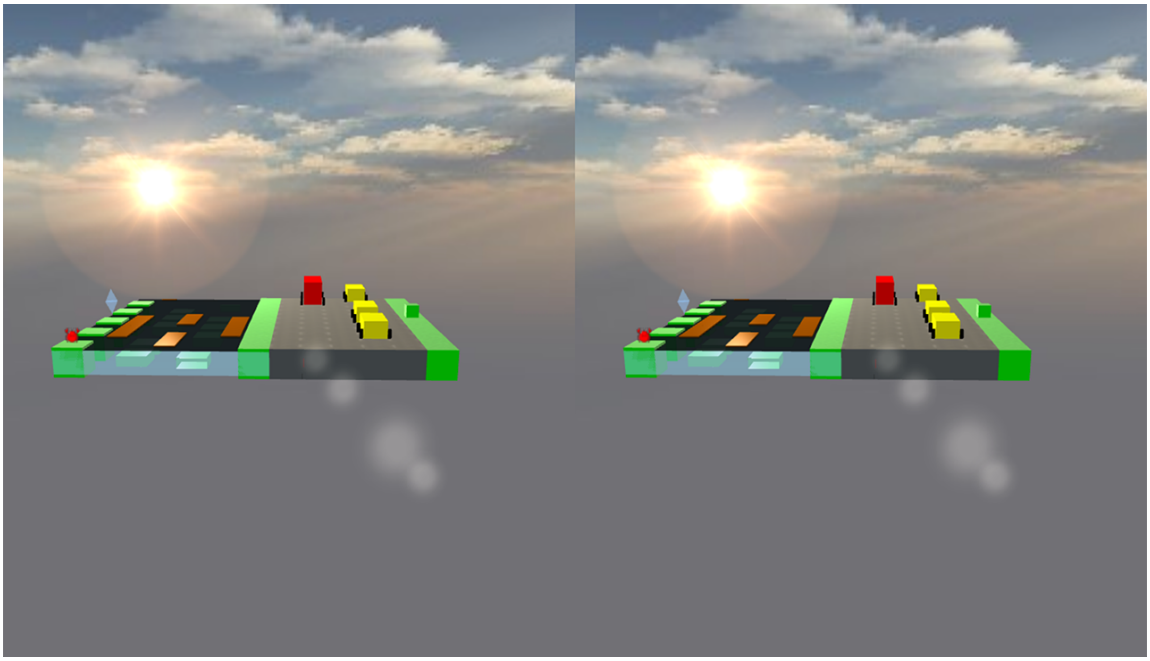

Figure 2: Stereo perspective rotating camera.

2 Object movement and collision detection

The objects position is after each timeframe by calling the update function.

Collision detection is done by checking the collision of the player and all of the collidable game objects each frame. ThreeJS makes this very easy as we can easily precalculate the bounding box for a given geometry and then easily check for collisions between them.

3 Lights

One of the lights can be seen on the Figure 1 and represents the sun. User can also toggle the point lights in the scene with the q key.

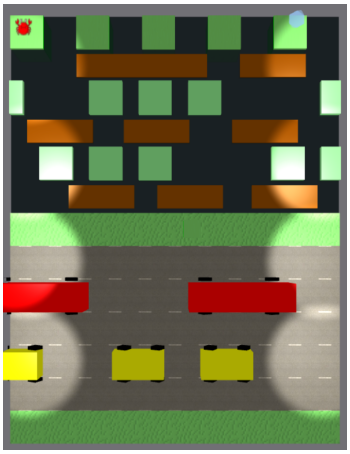

Figure 3: Pointlights activated.

4 Scoring system

Scene contains the current score on the top of the screen. The score is increased when the player reaches the target crystal on the other side of the river and the game also speeds up to get harder.

5 Fog

Creating fog in ThreeJS is done simply by using the THREE.Fog() class. User can toggle the fog on and off using the f key. The result is shown on the Figure 3.

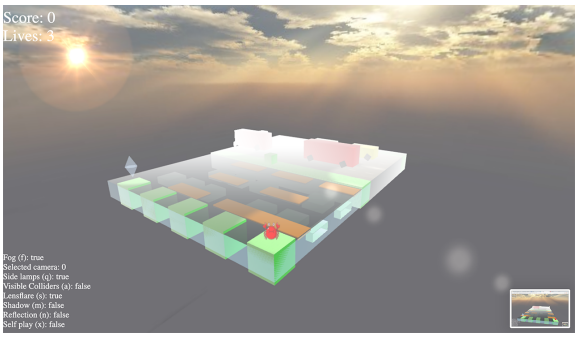

Figure 4: Fog.

6 Planar reflections and shadows

7 Billboard

We used a little crab to demonstrate the usage of billboards in our game. This can be seen for example on the Figure 3.

8 Lens flare

For our demo, we have utilized the ThreeJS Lensflare class, which makes the lens flares very easy. the lens flare is located at the "sun" in the sky and can be seen for example on the Figure 4.

9 Textures and skybox

ThreeJS allows the developers to write custom shaders for a material. The skybox is done using a large cube textured with a custom GLSL shader cube map.

We gave textured the ground with a grass texture as well as the road with the road stripes texture. Texturing using a GLSL shader was also utilized in the cubemap.

Conclusion

We have implemented most of the required functionality for the third task of our project. Sadly, due to time contains and also one of our group member missing, we have not implemented the bump-mapping and a particle system. The mobile gyroscope controls were developed, however we ran into a technical problem that the mobile browser is not able to access the gyro data over http and our hosting did not have SSL set up.

The testing was difficult, since Matyáš's old phone did not run any of the WebGL content, and Mykhaylo's phone was too large to fit into the Google Glass headset.