

# Three-Dimensional Vizualization and Animation

## Technical Report - Assignment III.

Marfeychuk, Mykhaylo  
ist194039

Skalický, Matyáš  
ist194904

### Introduction

The following document describes the work on the task 3, the development of the frogger game in the ThreeJS framework.

## 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.

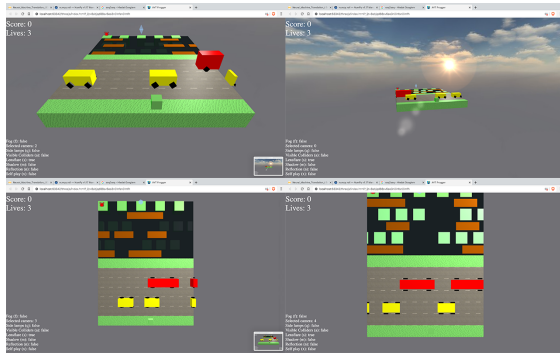


Figure 1: From top left: perspective following camera, rotating perspective camera, top perspective camera and top orthogonal camera.

The stereo viewing was implemented based on the module provided on the ThreeJS demo accessible at [https://threejs.org/examples/webgl\\_effects\\_stereo.html](https://threejs.org/examples/webgl_effects_stereo.html).

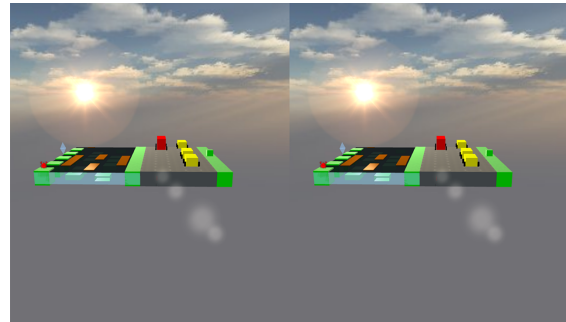


Figure 2: Stereo perspective rotating camera.

## 2 Object movement and collision detection

Each frame, we first check the collisions, update the game objects positions and then draw them to the screen.

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

The directional light can be seen on the Figure ???. The 8 point lights in the scene can be toggled with the usage of *q* key. We used the built-in ThreeJS light classes.

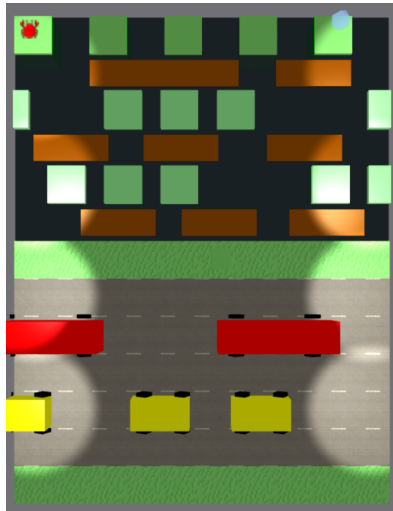


Figure 3: Pointlights activated.

## 4 Scoring system

Scene contains the current score on the top of the screen as shown on the Figure ???. 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.



Figure 4: Scoring.

## 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 ??.

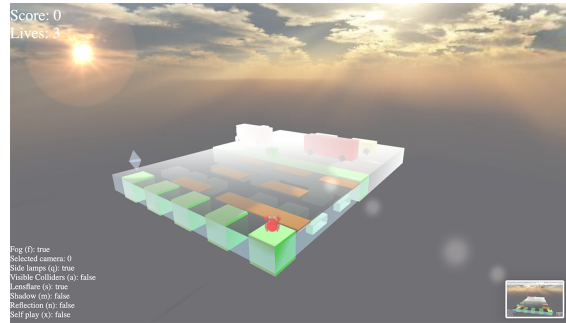


Figure 5: Fog.

## 6 Planar reflections and shadows

The planar reflection is implemented using a cube camera positioned on the other side of the plane which is then projected to the surface of the water. This effect, where the skybox is projected can be seen in the Figure ??.

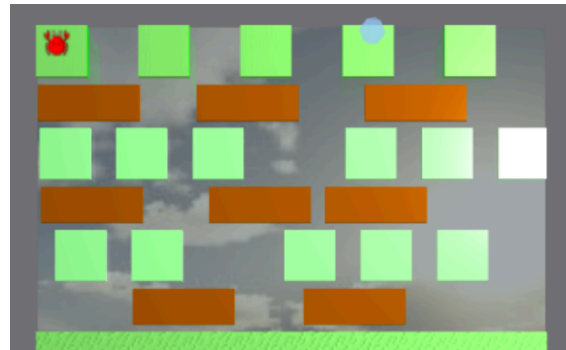


Figure 6: Reflection.

The shadows are created by using the native Three.js framework methods, which already does all the necessary to render the shadows. The framework first renders the models onto a plane, and then renders the real object on top. The shadows can be seen in Figure ??.

