2017-2018

Julio Ivan Davila Carrazco & Kyriakos Lite

Università Di ROMA “La Sapienza”

Interactive Graphics

A Polygonal Adventure

Contents

[Introduction 3](#_Toc491023202)

[Description of the Environment 4](#_Toc491023203)

[The Mainland 4](#_Toc491023204)

[The main character 4](#_Toc491023205)

[The Secondary characters 5](#_Toc491023206)

[Libraries, Tools and Models used in the project 6](#_Toc491023207)

[Tools 6](#_Toc491023208)

[Blender: 6](#_Toc491023209)

[Libraries 6](#_Toc491023210)

[Stemkoski: 6](#_Toc491023211)

[Three.js: 7](#_Toc491023212)

[Technical aspects of the Project 7](#_Toc491023213)

[The translation and rotation mechanism 7](#_Toc491023214)

[The follow mechanism 8](#_Toc491023215)

[Collision Detection 8](#_Toc491023216)

[Mission Success – Game Over mechanism 9](#_Toc491023217)

[Background Environment 9](#_Toc491023218)

[Implemented interactions 10](#_Toc491023219)

[Bibliography 11](#_Toc491023220)

# 

# Introduction

The “Polygonal Adventure” (figure 1) is a game made within the course of Interactive Graphics with purpose to realize the famous river crossing mathematical puzzle.

Within the environment, the user is expected to transport a wolf, a sheep and a cabbage across the river using a boat but there are certain constraints that the user has to take into account. To begin with the user can transport on the boat only one of the aforementioned characters at a time also if wolf is left alone with sheep, he eats it and if sheep is left alone with cabbage, he eats it too. There are many forms of this mathematical puzzle, usually different characters but their purpose is always the same to introduce one with the fact of mathematical constraints and persuade them to solve the given problem.



Figure 1 - Polygonal Adventure Interface.

# Description of the Environment

The environment and the game’s characters were created by us using tutorials and documentation about 3D modeling found on the internet. As it will be mentioned later in the document the environment used to realize the models is Blender.

## The Mainland

The game takes place at a floating island in space. As the name of the game suggests the environment is polygonal and can be seen below (figure 2):

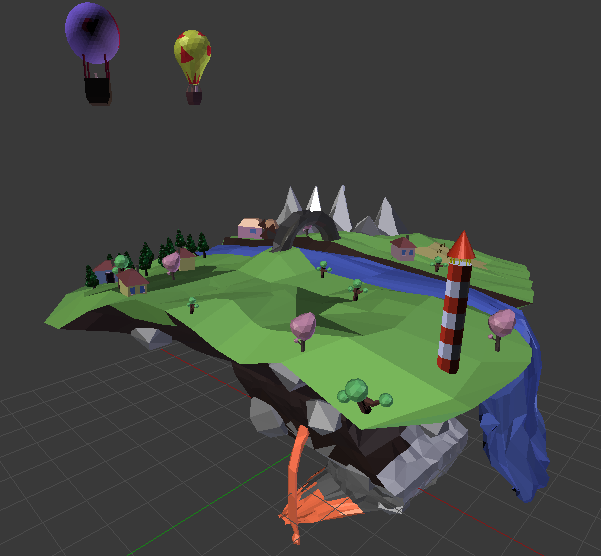


Figure 2 - Polygonal Island in Blender Environment

(LionStudiosTM, n.d.).

## The main character

The player is in control of the main character, this is the only character that can be control by the user but the other 3 characters (wolf, sheep and cabbage) can be move by certain actions of this character (figure 3).



Figure 3 - Main Character.

## 

## The Secondary characters

In this instance of the puzzle we decided to use a wolf, a sheep and a cabbage (table 1) as the characters that the human (main character) will control, this characters are loaded on one side of the island at the beginning of the game and the player has to guide them to the other side meanwhile taking into account the several constraints impose by the puzzle.

|  |  |  |
| --- | --- | --- |
| **Table 1 - Secondary Characters** | | |
| **Wolf** | **Sheep** | **Cabbage** |
| Figure 4The Wolf in Blender Environment | Figure 5 The Sheep in Blender Environment | Figure 6 The Cabbage in Blender Environment |

# 

# Libraries, Tools used in the project

There are a few and very significant libraries and tools used for the completion of the project, as far as for models as we already mention everything was developed by us. The in depth explanation of them exceeds the purposes of this document and we will state them bellow with a small description.

## Tools

### Blender:

Blender (figure 7) is a widely used free and open source 3D development suite. It supports the entirety of the 3D pipeline needed by a developer to develop a 3 dimensional model, rigging, animation and many more functionalities.

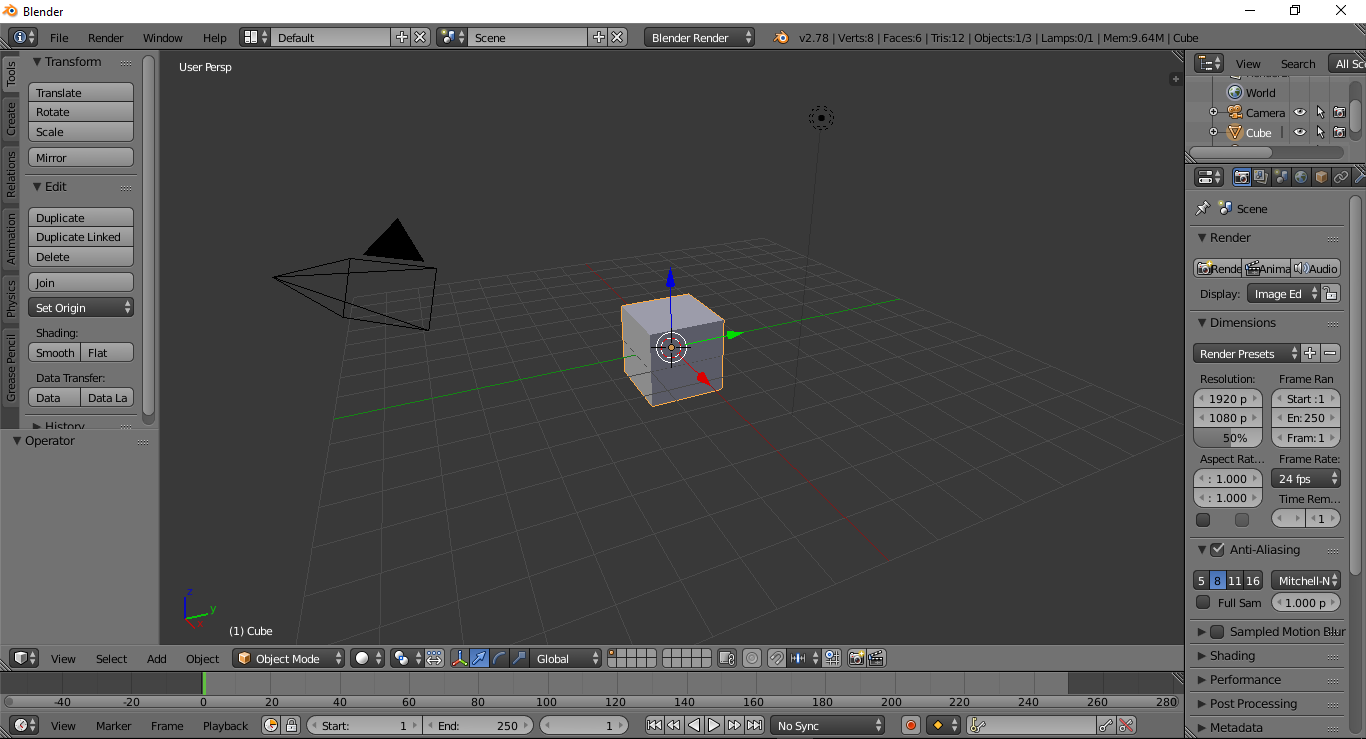


Figure 7 - Starting Interface of Blender

In our case, blender was used to create the mainland and the characters from scratch. In the same environment we did the rigging (imposing armature to the mesh in order to manipulate it) and create the walking animations.

(Blender Foundation, n.d.)

## Libraries

### Stemkoski:

The stemkoski library is a github repository of tutorials with a goal to provide basic and instructive examples of various features in three.js and it belongs to a mathematics professor and computer science Lee Stemkoski. The examples are very instructive and understandable a fact that helped us greatly in realizing our project. From his tutorials mostly the skybox, the information box and collision detection were used.

(Stemkoski, n.d.)

### 

### Three.js:

This library is a lightweight cross-browser JavaScript library/API used to create and display animated 3D computer graphics on a Web browser. Three.js scripts may be used in conjunction with the HTML5 canvas element, SVG or WebGL.

Three.js allows the creation of GPU-accelerated 3D animations using the JavaScript language as part of a website without relying on proprietary browser plugins. This is possible thanks to the advent of WebGL. The library can render using Canvas, SVG and WebGL.

(threejs, n.d.)

# Technical aspects of the Project

There are a lot of functions implemented in order to realize this project but we choose the ones that are very essential for the competition of the project and stated the bellow.

## The translation and rotation mechanism

In order for the main character to navigate through the terrain, he is translated when W for forward and S for backwards are pressed but also rotated around his longitudinal axis. The relevant code is shown below (figure 8).

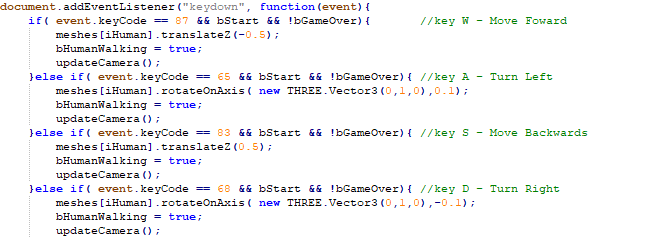


Figure - The translation rotation mechanism

**Note!** The camera always follows the main character from behind as a first person shooter game thus its updated along with the movements.

## The follow mechanism

The follow mechanism, exists in the eventListener.js file and works in a very comprehendible way. When one of the ‘z’, ‘x’ or ‘c’ keyboard keys are pressed the “calculateDistance” function is called which measures the distance between the main character and the character that corresponds to every key. If the distance is smaller than a certain threshold then the selected character is assigned an exact distance from the main character and follows him whenever he goes.

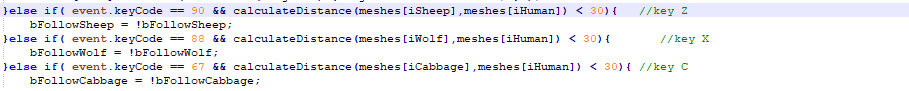


Figure - The follow mechanism activator – deactivator

**Note!** The calculated distance is the Euclidean distance between 2 points in 3D space given by the equation:

## Collision Detection

The collision detection is very essential for a virtual environment such as a game because it provides the necessary realism by obeying the physical rules of the real world.

Every time the characters move, there is a check in the event listener file that calls the function “detectionCollision”.

The meshes that are collideable (such as trees, houses, etc.) are assigned the “Box3” which is an invisible bounding box that contains the mesh (figure 10), if there is an intersection between bounding boxes then the character is prevented to move more towards the collision.

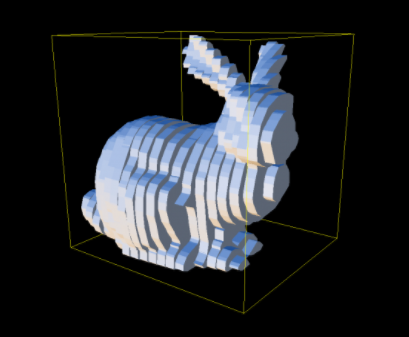


Figure 10 - The bounding box can be visualized with yellow color.

More specifically, in order for the characters not to fall off the island or in the river, planes without material (which makes them invisible) were added to the environment with the bounding box attribute. Again, if there is an intersection calculated the character is prohibited to move towards that direction.

## Mission Success – Game Over mechanism

As we know the purpose of this puzzle is to move the animals from the one side of the river to the other keeping in mind the constraints established. At every moment the distance between the constrained animals and the human is calculated. If for example the wolf and the sheep are in dangerous range between them and the human is departed more than a specified range then GAME OVER pops in the window and the player has to restart the game.

For the mission to be successful the animals have to be together under a specified radius near to the farm house across the river.

A good tactics for the game is to run around in the beginning and collect all the animals in front of the raft, and then cross them in the right sequence.

## Background Environment

In order to visualize the mainland into space we used a skybox as it’s called, which is nothing more than a box containing the mainland whose 6 faces are assigned a desired texture. It is a very efficient and realist way to realize a 3D vast environment. The skybox textures were picked from (Custom Map Makers, n.d.).

# Implemented interactions

The possible interactions that the user can apply is through the keyboard as follows:

|  |  |
| --- | --- |
| **Main Character Controls** | |
| **Movement Controls** | |
|  | Keyboard keys used to navigate the character:  **W:** Forward,  **L:** left, **S:** Backwards, **R:** Right |
| **Game Play Commands** | |
|  | **Z,X,C :** Activation keys for follow and unfollow (applicable only at a minimum range away of the secondary characters) |
|  | **B:** keyboard key used to board – deboard the character on the raft |

Table 1Interaction Controls

Also, the user can click on the top left icon as shown in Figure 1 which pops up an information window that explains the purpose of the game as follows.

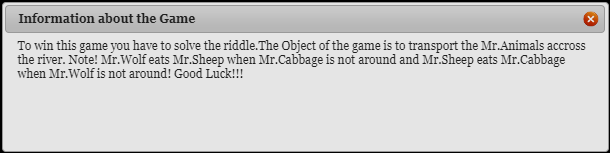


Figure 11Pop up box

# Bibliography

(n.d.). Retrieved from Blender Foundation: https://www.blender.org/

(n.d.). Retrieved from Custom Map Makers: http://www.custommapmakers.org/skyboxes.php

(n.d.). Retrieved from LionStudiosTM: https://www.youtube.com/user/xVxMARKxVx/featured

Stemkoski, L. (n.d.). Retrieved from stemkoski: http://stemkoski.github.io/Three.js/index.html

*threejs*. (n.d.). Retrieved from https://threejs.org/