First Person Shooter Tank

Pirouz Sourati Zanjani
Computer science department
2017-2018

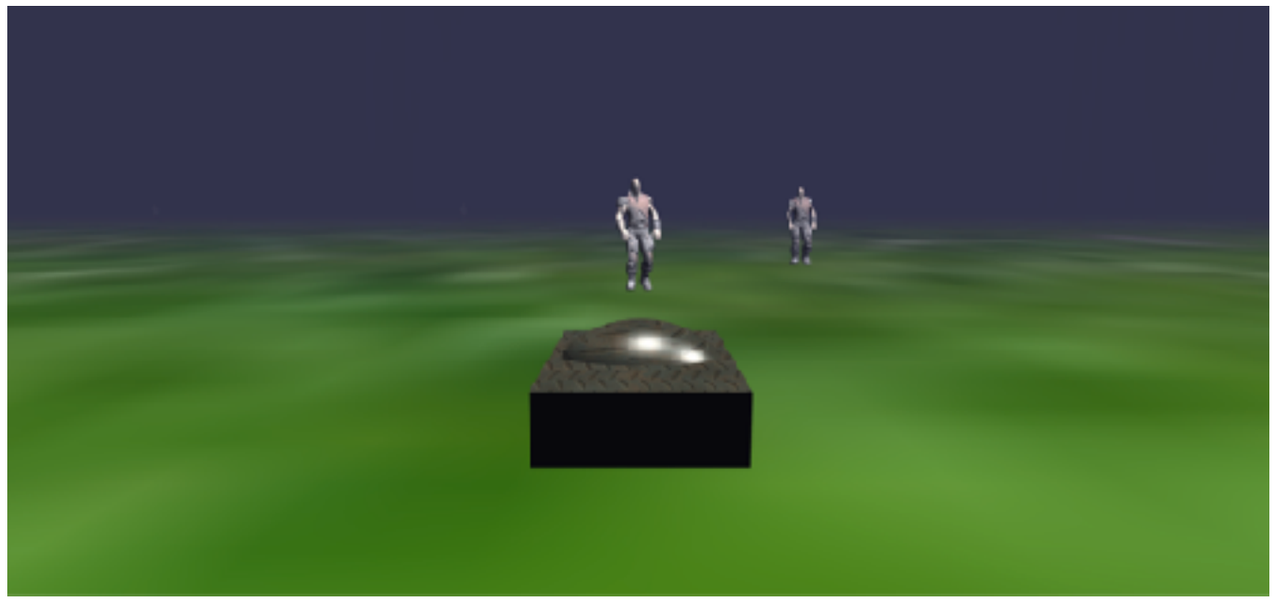
Sapienza
Università di Roma

- **Introduction**
- **Libraries**
- **Technical aspects**
- **Interactions**

Introduction

First person shooting tank is a webgl based game using Babylon js. In this game user can shoot and eliminate (or kill) the enemies that are attacking and following him.

User is going to able control the tank with “WASD” keys and shoot to the targets with the “B” key.



In this game models and animations implemented with Babylon js and physics applied to necessary objects in the scene.

This project consists of complex model which are the enemies, following camera, lights and textures, defined keys for user interaction and animation.

For the purpose of simplicity all the objects and models belong the one scene and enemies were cloned from the first model which were imported to the project.

Libraries

Libraries used in this project are:

Babylon JS :

A framework for building 3D games with HTML 5 and WebGL. This game uses Babylon js core engine in order to crate meshes, lights and …

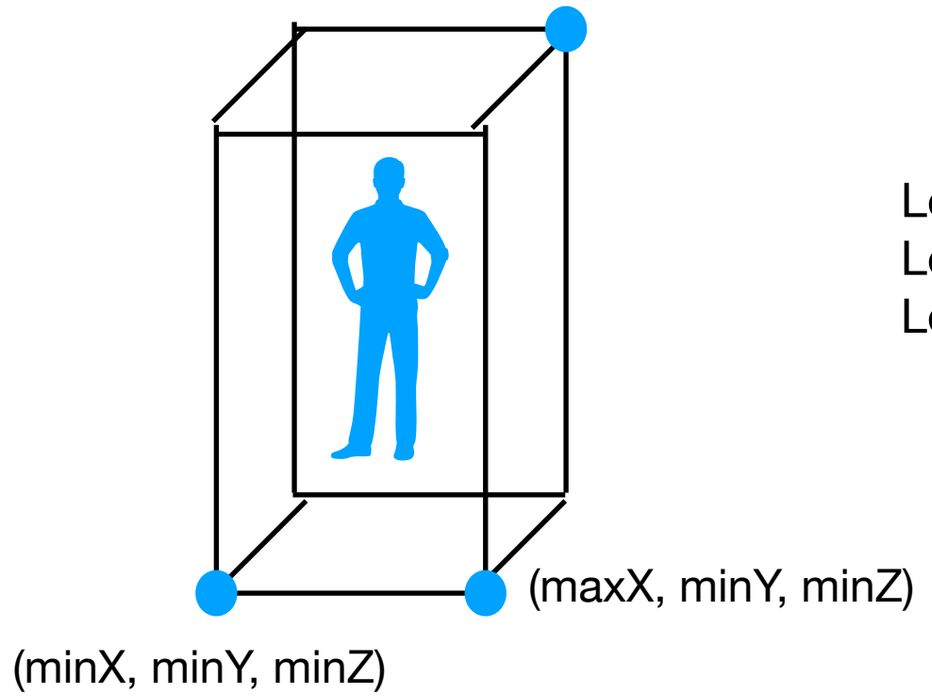
Cannon JS:

Cannon.js is an open source JavaScript 3D physics engine. Unlike physics engine libraries ported from C++ to Javascript, cannon.js is written in Javascript from the start and can take advantage of its features.
In this game physics applied to some objects like the ground and or the behaviour of the firing ball was get form the cannon js library.

Technical Aspects

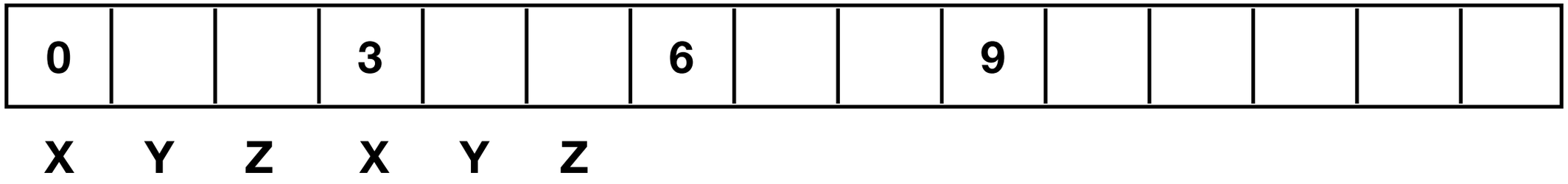
Avoiding the aligning of the models:

After cloning the models I realised that after cloned models (enemies) are merging and aligning in the same position to prevent this behaviour blinding box was created for each model



Length X = maxX - minX
Length Y = maxY - min Y
Length Z = maxZ - minZ

Babylon JS provides a function to return the points as an array like above



For example for the array like a = [10,8,5,20,3,5] we can first assume minX is a very big number like 99999 and then:

```
For (var i=0 ; I < a.lenght ; i++){
   If (minX > a[i])
      minX = a[i]
      }
```

after calculating this points we can create a bounding box for each model and use bounder.moveWithCollisions in order to prevent aligning the models.

ActionManager:

Actions are a simple way to add interactions in your scenes. An action is launched when its trigger is fired.

For instance, you can specify that when the user clicks (or touches) a mesh, an action is executed.

Action manager was used to specify when a cannon ball hits the model. The trigger I used was

OnIntersectionEnterTrigger. This trigger raised when the mesh is in intersection with a specific mesh and Raised just once.

```
cannonBall.actionManager = new BABYLON.ActionManager(scene);
scene.dudes.forEach(function (dude) {
    cannonBall.actionManager.registerAction(new BABYLON.ExecuteCodeAction(
        {trigger : BABYLON.ActionManager.OnIntersectionEnterTrigger,
        parameter : dude.Dude.bounder },
        function () {
            dude.Dude.bounder.dispose();
            dude.dispose();
        }
    ));

});
```

The End