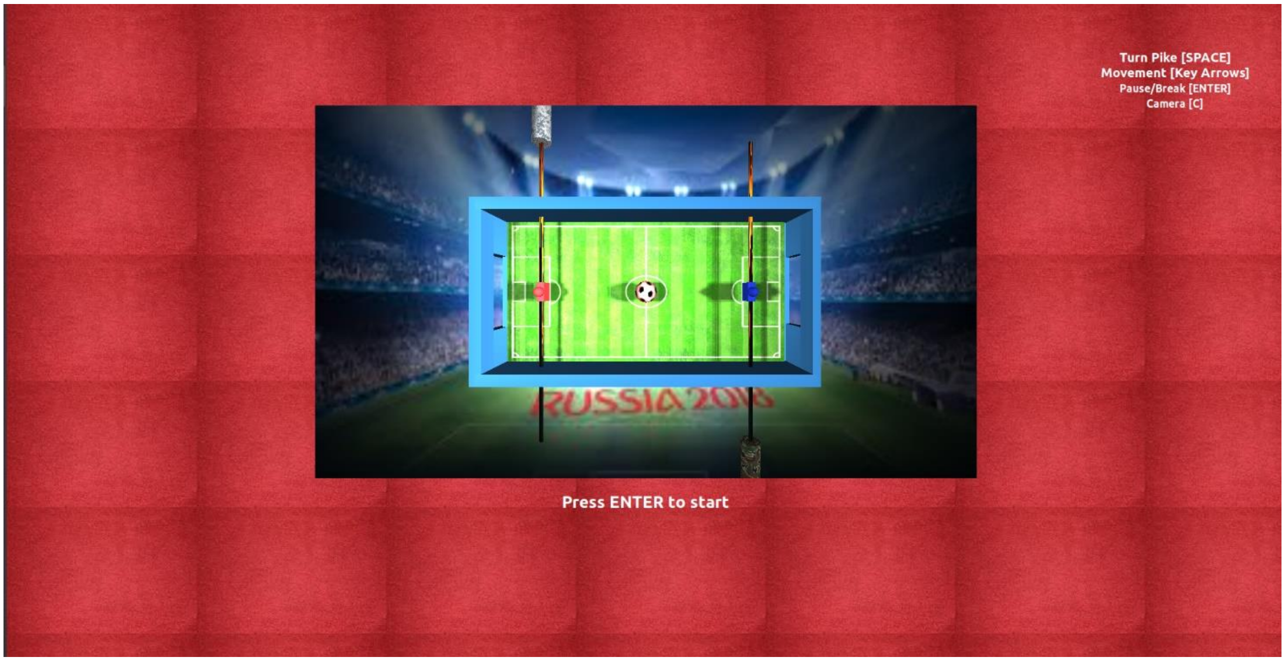
SAPIENZA
UNIVERSITÀ DI ROMA

Interactive Graphics
Project Report

Project Name: Foosball Game

Submitted To: Prof. Shaerf Marco



Group Members:

Zubair Baqai	1849940
Arslan Arshad	1861810

Table of Contents

Foosball Game

Introduction:

This game project is based on the popular table game Foosball(Table football). A standard foosball game included twenty two number of players but in our case, we tried to make its simpler version by just adding two players. We can control our player through arrow keys and hit the ball by rolling our rod through space key.

Game Environment:

For this project we are using three.js which is most widely used to render 3D computer graphics for web browsers. While in the course we studied WebGL we used three.js because it is the most similar version as of WebGL. Majority of the geometries like Plane Geometry, Cube Geometry, Cylinder Geometry, Spherical Geometry and lastly self-defined Object 3D Geometry. Our game environment is divided into several parts.

- Canvas Background
- Game Board
- Playground
- Camera
- Light Source

Canvas Background:

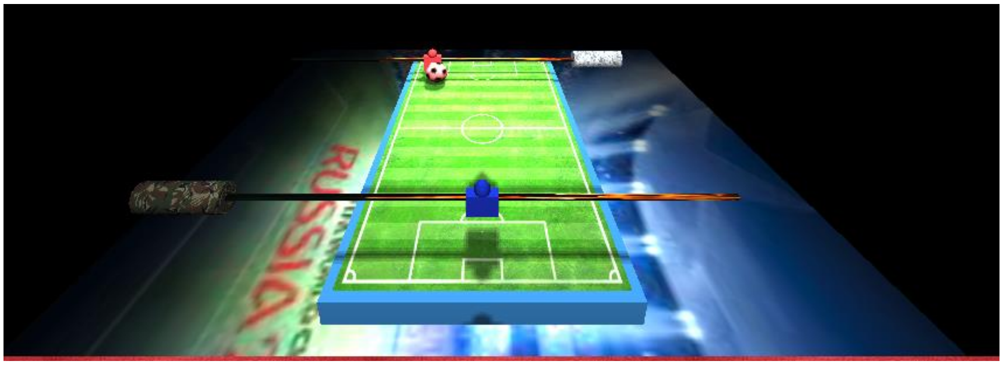
Background is a plane geometry with a negative z-axis value. This negative value will help us to set our texture as a background.

Game Board:

The board consists of a smooth surface with the playground texture, and a few walls made with a cube. The walls include two big walls and two small walls. Both big walls have attached rods to control player position to hit the ball. There are two goal gates made on both sides of the table by adding small cubes along with the small walls. Because of these small cubes, there will an empty area through which we can decide who is the winner.

Playground:

The playground of the game lies on the center of the board. We added a rectangular shape football ground texture image. This is the main component of our game, all functions calls are according to the keyboard keys or ball movement on the playground, for which we check status of our game after every six frames per seconds.



Walls:

As mentioned above in the description of Game Board we have eight cubic geometry to show the walls around our playground. Whenever, ball touches these wall it will move in the reverse movement.

Rods:

Rods are made up of cylindrical geometry both rods divided on two parts according to their radius. A part having small radius and larger length is the main part. And a part having a larger radius and smaller length is called gripper which is placed at the end of the rod with a different texture.

Ball:

Ball is a spherical geometry which is the main part of our project. Every position of the ball trigger the functionality respectively to get the output.

Camera:

In the environment we used perspective camera and the view angle is 75 with the near and far values of 0.1 and10000. There are two cases for the rotation of our camera. First one where x=0, y=0 and z=-180 with the position of (0,0,350) it means camera is looking downward toward the table(Gameboard). And the Second one where x=0, y=-70 and z=270 with the position of (-400,0,250) it will give us the side view from the players side.

Light Sources:

There are multiple light sources in this game one point light and three spot lights. Point Light is use to illuminate all directions. While one spot light pointed toward the opponent(computer), second spot light pointed toward the player and third one which illuminate from the center of the scene towards the opponent(computer) goal.

Game Physics:

Through this description we can understand the game more precisely. Physics of our game is divided into the following parts.

Rod Movements:

Whenever key is pressed on the keyboard rod will move with the speed of 2 pixels per frame rate. There is also an autonomous rod movement for Opponent(Computer) according to the Ball Movements. So, we can say that rod movements is further separated in the following categories.

- User Rod Movements
- Opponent(Computer) Rod Movements

Ball Movements:

Balls is moving with an initial speed of 1pixel(1x,1y) on our playground. Whenever we press Space key it will increase by 2pixel(2x,2y). there are three different types of ball movements. In the case of rebound movement y will be inverse according to the target wall.

- Normal Movement
- Rebound Movement
- Player Shoot Movement

When player shoot the ball the direction will be decided on the basis of their player geometric angle.

Goal Scoring

Whenever ball crosses the gate(gx, gy) positions then It will consider it as a goal and game reset itself to its initial state.

Controls:

Table given below shows the keys along with their functionalities.

Key	**Functionality**
Right/Up arrow key	Move the rod right/Up
Left/Down arrow key	Move the rod left/Down
Space	Hit the Ball which will increase its speed
C	To change the position of the Camera
Enter	For the Pause and Play purpose

Libraries:

The following libraries were used in the project:

- **ThreeJs**: This was used to create and display animated 3D computer graphics in the web browser.
- **JQuery**: This was used to perform HTML DOM Tree Traversal and Manipulation.
- **KeyboardJs**: This was used to bind keys to perform actions on the scene

How to Run:

Make sure you have installed correctly **node.js** and **npm**. After installation of both tools, you need to use http-server command to run your code in the browser.

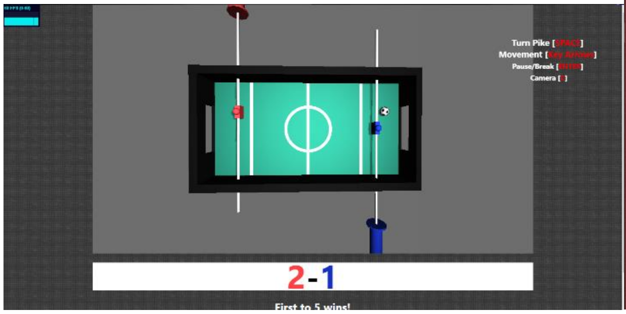
Our Contribution:

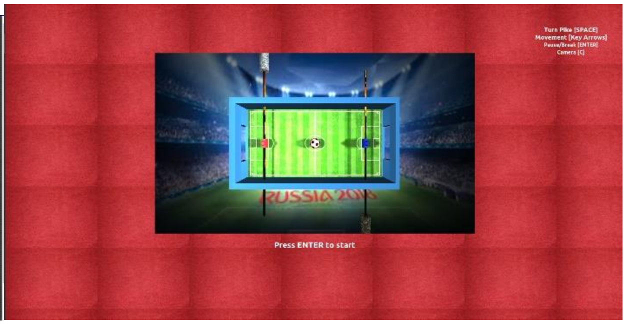
After understanding the concept of the reference code we understood it line by line. We made many changes and added some new features in it. Some of them are listed below:

- Background Image
- Board wall colors
- Texture Modification on playground, rods and griper
- Light source addition to make its clear view
- Self-defined static camera positions
- Movements of balls(Rotations of x and y of balls have been modified according to be more realistic physics)
- Removal of score
- Game reset option
- Player hitting functionality(whenever player use space while hitting the ball it will move faster with more power)

- Added two different sounds on the startup and on the goal event
- Shadow qualities have been improved
- Orientation of the lightning sources based on the camera Mode

Moreover, previously there was a bug in the game that when we hit the ball with a moving rod the ball speed increased with a very non predictable manner but now it will move smoothly





References:

1. https://github.com/amingholami/WebGL-mini-Football
2. https://threejs.org/