Bouncing Ball Simulation

Introduction

This project showcases a Bouncing Ball simulation, designed to illustrate basic physics principles and game development techniques. Developed with the OpenGL and GLUT frameworks, it offers a dynamic visualization of a bouncing ball within a 3D environment, providing insights into real-time rendering and physics-based animations.

Features

- **Physics-Based Motion**: Simulates realistic bouncing behavior, incorporating gravity, elasticity, and collisions.
- **Interactive Controls**: Users can adjust various parameters such as gravity magnitude, ball elasticity, and surface friction to observe different behaviors.
- **3D Camera Manipulation**: The simulation includes interactive 3D camera controls, allowing viewers to explore the scene from multiple perspectives.

Getting Started

Prerequisites

To ensure a smooth experience, please install the following tools:

- A C++ Compiler (GCC or similar)
- OpenGL Libraries
- GLUT Framework

Building and Running the Simulation

1. Clone the repository to your local machine:

```
git clone https://github.com/AndreeaDraghici/Bouncing-Balls-3D.git
```

2. Navigate to the project directory:

```
cd Bouncing-Balls-3D
```

3.Compile the code (example given for g++):

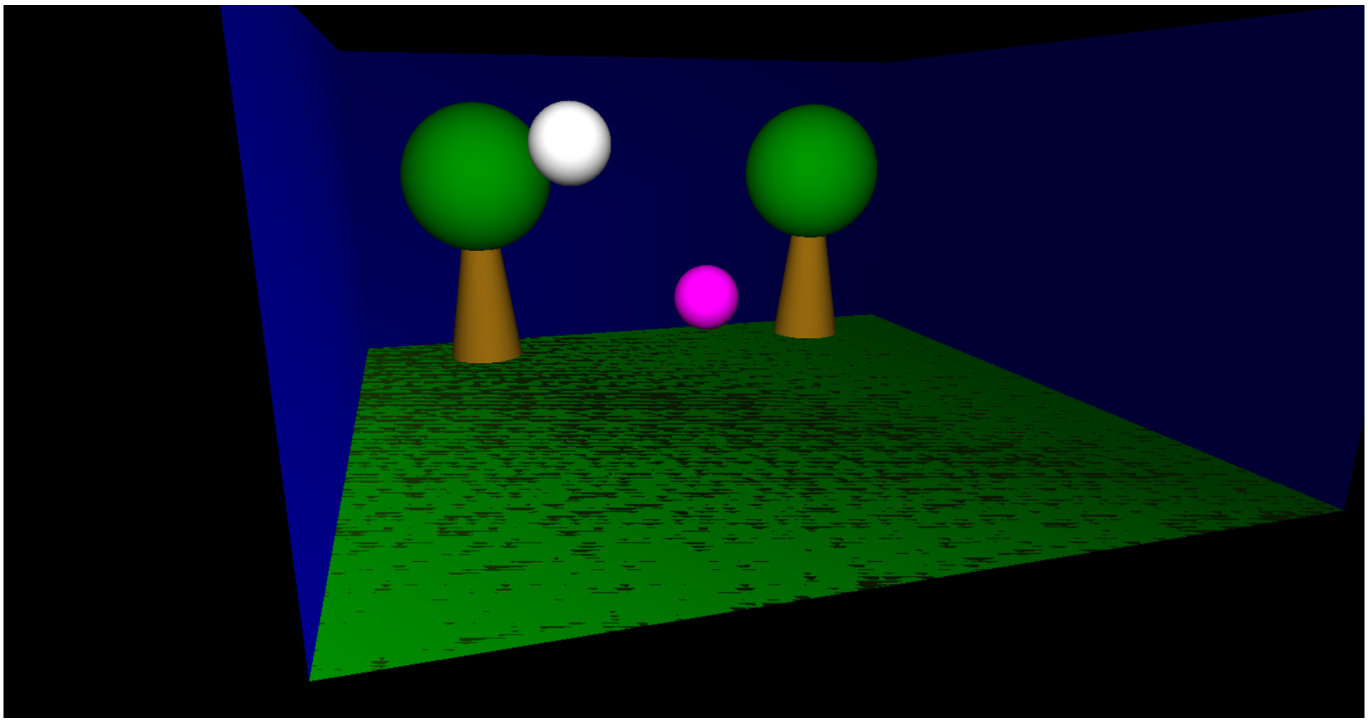
```
g++ -o Bouncing-Balls-3D Application.cpp -lGL -lGLU -lglut
```

4.Run the simulation:

```
./Bouncing-Balls-3D
```

Development Environment

The simulation was developed on a Windows 10 machine, using Visual Studio 2022, and is written in C++ for cross-platform compatibility. It uses the OpenGL library for rendering graphics and the GLUT framework for managing the GUI and handling input events.



Customization

To customize the simulation, you can modify the source code to change parameters such as the initial position and velocity of the ball, the elasticity coefficient, and the simulation speed.

License

This project is licensed under the MIT License - see the LICENSE.md file for details.

Usage

Upon launching the simulation, you will see a ball bouncing within a confined space. Use the keyboard to pan and tilt the camera around the scene.

Rendering Techniques & Physics Engine

OpenGL functions are used to draw the scene, including the ball, ground, and surrounding walls. Lighting effects are added to enhance realism. The simulation calculates the ball's trajectory using the equation of motion, considering gravity and energy loss on impact.