E12: Physics Solver

Course: IGME 309 – Real Time Simulations for Games II

Golisano College of Computing and Information Sciences

School of Interactive Games and Media

Rochester Institute of Technology

Due: Check in MyCourses

Deliverable: Single zip file containing SolverExt.h, SolverExt.cpp and AppClass.cpp

**Objective:**

The objective of this exercise is for students to explore and manipulate the various parameters of a physics solver class to fine-tune the behavior of objects in a video game simulation. Through adjusting key values such as gravity, repelling forces, friction, and other physical attributes, students will gain a deeper understanding of how physics engines work and how to manage and control the dynamics of characters in 3D space. The exercise emphasizes the importance of understanding the limitations and behaviors of the physics engine being used and demonstrates how careful adjustments can achieve a range of realistic and responsive behaviors.

By completing this exercise, students will:

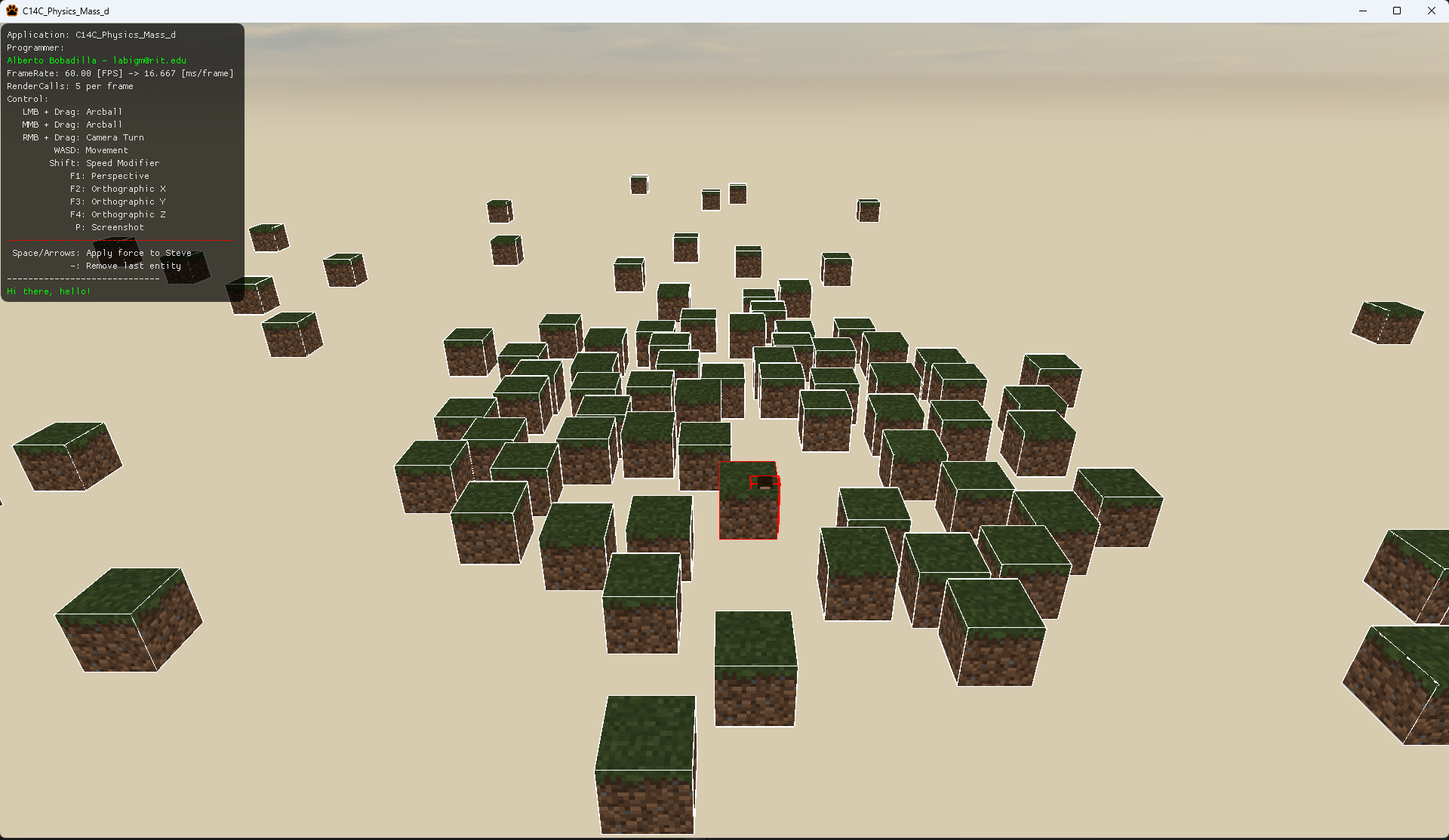
* Learn how to work with a physics solver class, adjusting parameters to achieve the desired physical behaviors for characters and objects within a simulation.
* Understand how different forces (such as gravity, friction, and repelling forces) interact to create realistic movement and prevent characters from falling through platforms.
* Gain practical experience with fine-tuning the solver to handle real-time physics simulations, emphasizing the trade-offs between precision and performance in game development.
* Explore the concept of **"tuning"** in game physics and how modifying solver parameters can change gameplay, player experience, and physical realism.
* Appreciate the **limitations of physics engines**, learning how different engines may have strengths and weaknesses and how to overcome these challenges through parameter adjustments.
* Develop problem-solving skills by implementing a solution to the problem of preventing a character from falling through platforms, with the flexibility to explore various approaches to the challenge.

This exercise offers students the freedom to experiment with different approaches and techniques, fostering creativity while reinforcing the understanding that physics-based simulations require careful adjustments to achieve the desired results. It serves as an opportunity to learn how different simulation parameters can be used to influence both the realism and gameplay mechanics of a physics engine.

**Instructions:**

This exercise follows lecture D14

1. Your code will generate a solution really similar to C14C like this:



Steve pierces the cube and then once he reaches the base he separates, this is wrong.

1. This exercise will include the External Solver found in C14D
2. Your exercise is to fine-tune the values sent to the solver or modify the methods in the external solver so Steve does not pierce the cubes as he is falling and stay on top of them.
3. You may work in teams for this exercise but each student needs their individual solution.
4. Your submission will be only 3 files, SolverExt.h, SolverExt.cpp, AppClass.cpp please zip these files
   1. Note, not all the files might have been modified, this is based on the way you solve this exercise, even if you didn’t modify one or more of those files, please include them.