COMP 438 - Winter 2025

Project Report

Marie Giacomel 40321702

Description

The project is developed in C++ using libigl and focuses on simulating various 3D meshes, including cubes, icosahedrons, pyramids, pentagons, cones, and cylinders. Those primitives behave like jelly-like soft bodies.

Additionally, users can import custom models by specifying the path to a .obj or .off file. Once the viewer is launched, users can control the simulation through a menu developed with ImGui, which provides options to start and pause the simulation, reset the scene, modify the model's height, and switch between available models. The viewer also supports camera movement, allowing users to rotate the view and navigate the scene freely.

When the simulation starts, the selected model falls under gravity, and the mass-spring system is applied to each edge to compute spring forces. This simulates a jelly-like deformation, making the model bounce and deform upon impact with the ground in a realistic manner.

What was/wasn't accomplished

The project took a significant amount of time to develop, and I encountered numerous challenges along the way. Some of these difficulties were expected, while others were unforeseen or more complex than anticipated. Sometimes, I had to make compromises and adapt myself, either by finding creative solutions or by slightly redefining the original objectives. However, despite these challenges, all the objectives outlined in the project proposal were either fully or partially achieved, and the final project is complete.

The main objectives were to implement a mass-spring system for soft-body simulation, generate primitives meshes, develop a user interface to control the simulation, and allow custom model importation.

The project successfully supports multiple 3D primitives such as cubes, icosahedrons, pyramids, pentagons, cones, and cylinders. The mass-spring system enables realistic jelly-like deformation upon impact with the ground. The ImGui-based interface provides intuitive controls for managing the simulation, including starting, pausing, resetting, adjusting model height, and switching between available models. Additionally, custom .obj and .off models can be loaded into the simulation by adding the path after the executable command. Despite the challenges faced during development, the project effectively demonstrates real-time soft-body simulation.

What was/wasn't difficult

Throughout the development of this project, I encountered several challenges, some of which were expected, while others required creative solutions or adjustments to my initial plans. Below are the main difficulties I faced:

- **Usage of a global structure**: I needed a way to maintain global data without passing multiple parameters across functions. Finding the right approach took time and experimentation in order to keep a clean code structure that would allow me to easily add features and continue my developpement. The singleton design pattern was the right option in my case Even if I needed to rework it twice before my final version.

- **.Obj files made with Blender**: I try to use Blender to create my own models to test them in my Jelly Simulator, but the .obj file generated by Blender contains texture coordinates and normal information stored on faces, and libigl's .obj parser does not support this format. To work around the problems and correctly load the custom models, I then created a python script allowing me to delete unnecessary data and correctly reformat the file.

- **Setup the project**: Initially, I tried to manually create the CMakeList file but I couldn't figure how to make it work. My first attempt was on Windows, like during the lab session. But after explaining my issues to the TA, he advised me to try with Ubuntu, and to reused the configuration of the tutorial example. But even after doing that, the compilation process was very slow. After some research, I discovered that using Ninja with the -J flag significantly improves build times.

- **Usage of ImGui's Gizmo tool**: I spent several days trying to integrate ImGui's Gizmo but couldn't get it to work. After multiple failed attempts, I decided to use keyboard, mouse inputs, and a user menu to make the simulation interactive instead of the Gizmo I originally wanted to use.

- **Unstable simulation parameters**: Finding the right values for stiffness, damping, and speed was difficult because the simulation was highly sensitive to small changes and was heavily dependent on the model used. I first wanted to allow the user to adjust these parameters but it would likely make the simulation unstable and negatively impact the experience. I preferred to keep them as constants but suggested more model examples.

- **Time constraints**: Even if the deadline was at the end of the term, I took too much time to choose and start the project and probably underestimated the time needed for this project. My optimism led me to procrastinate on this project, thinking I could complete it in less time than I actually needed. As a result, I focused too much on other assignments and projects that were due earlier. In the end, balancing this project with others during the end-of-term rush was challenging, I struggled to find enough time to work efficiently, which added extra pressure during the development.

- **Mass-spring system implementation**: Surprisingly, I didn't have too much trouble implementing the mass-spring system. Perhaps this was because it was the part I was the most scared about, so I anticipated the challenges and prepared well by researching detailed explanations of how it works.

- **Argument parsing / Menu**: These parts were the easiest since I was already familiar with C++ and had previous experience with ImGui. Because I understood how it works, I was able to create the menu and parse the model path quickly enough to avoid any impact on my development schedule for this project.

Personal comments

Even if the final project doesn't look exactly like what I was planning to do, I'm satisfied with my work, considering the different challenges I needed to face. This project enabled me to discover Libigl, learn how to use this library and better understand how the soft body component in game engine works. This will be useful for my end-of-study project, as well as furthering my knowledge of computer graphics. I hope This project fully fills what was expected for this course and shows my investment and that I put efforts into this project.