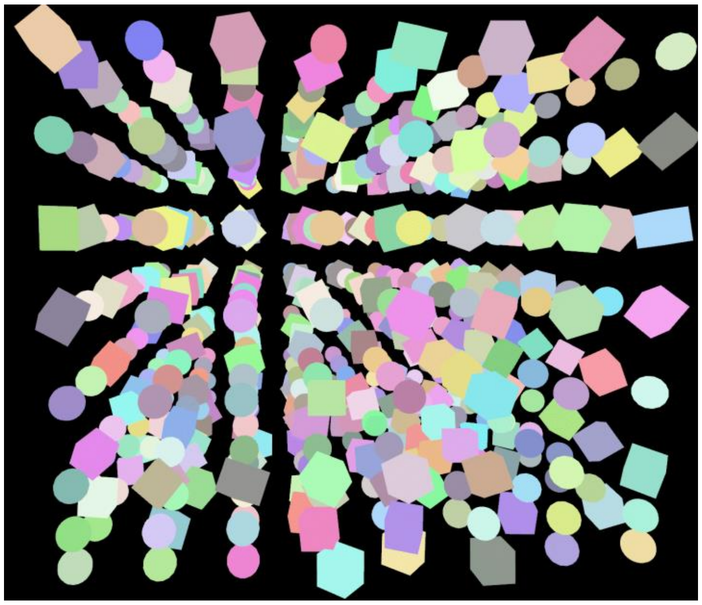
Applications

The following application demos have been included to represent the capabilities of the engine.

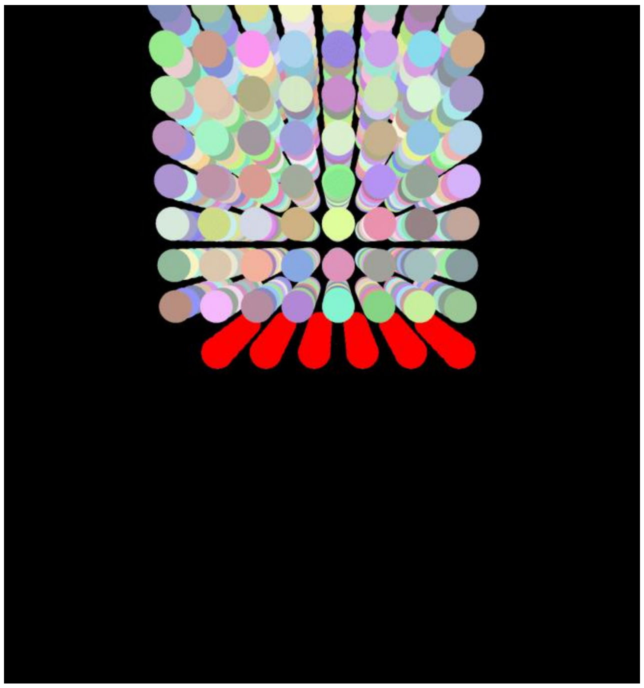
Tower Demo

A 3D grid of rigid body objects evenly spaced apart, either a cube or sphere selected randomly upon startup.



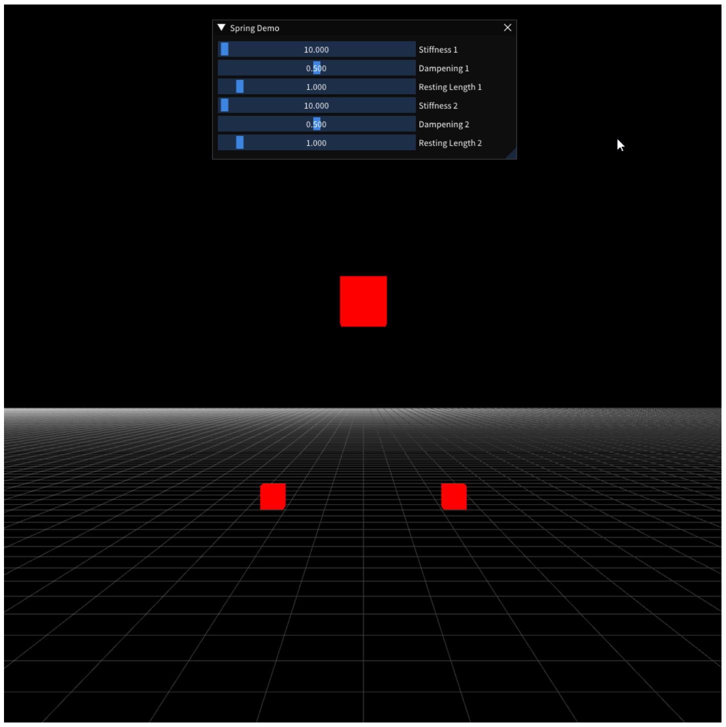
Particle Demo

Showcasing particles, a type of physics object used for springs, cloths and other constraints.



Spring Demo

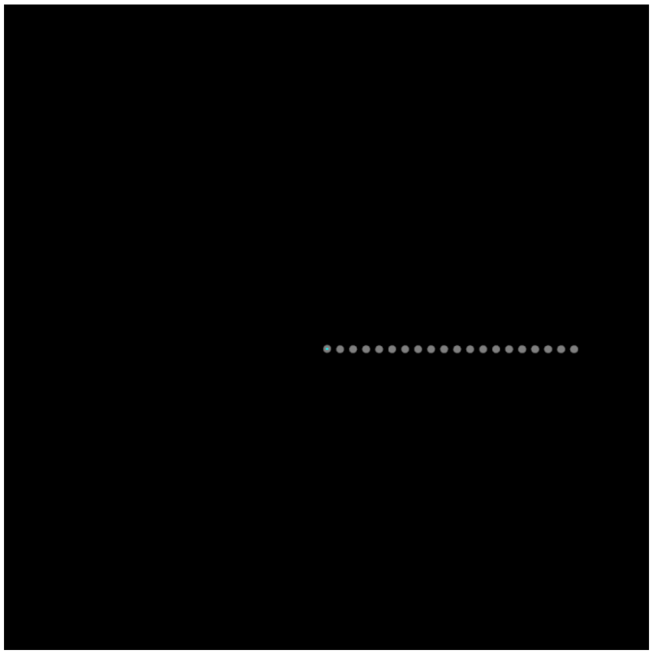
Two springs attached to a static object, accompanied by parameters to show how they affect the springs.



```
10.000   Stiffness 1
0.500    Dampening 1
1.000    Resting Length 1
10.000   Stiffness 2
0.500    Dampening 2
1.000    Resting Length 2
```

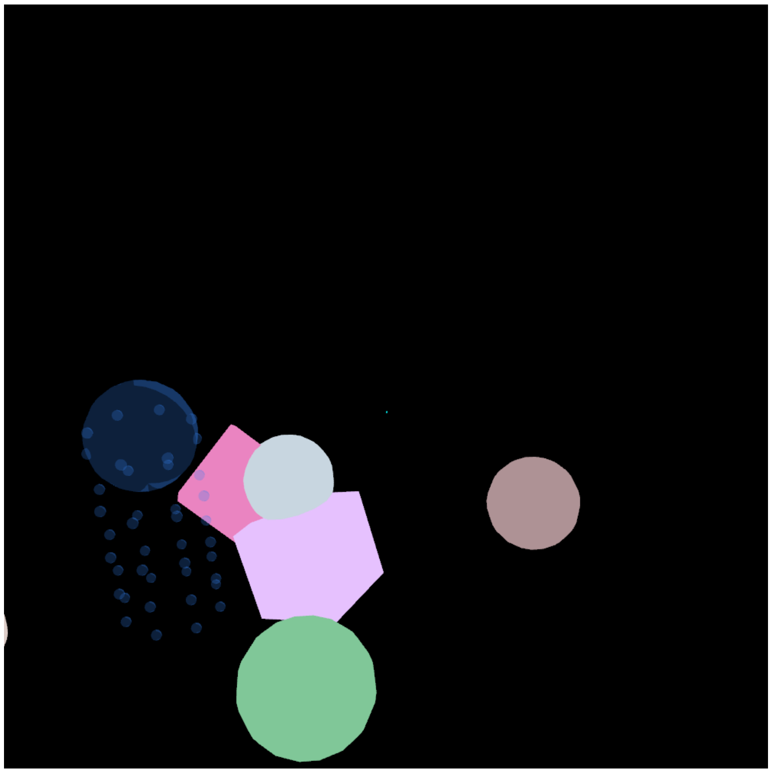
Rope Demo

A chain of springs attached to a static point. The gif below shows interaction with the floor, and reaction to a force applied by the camera.



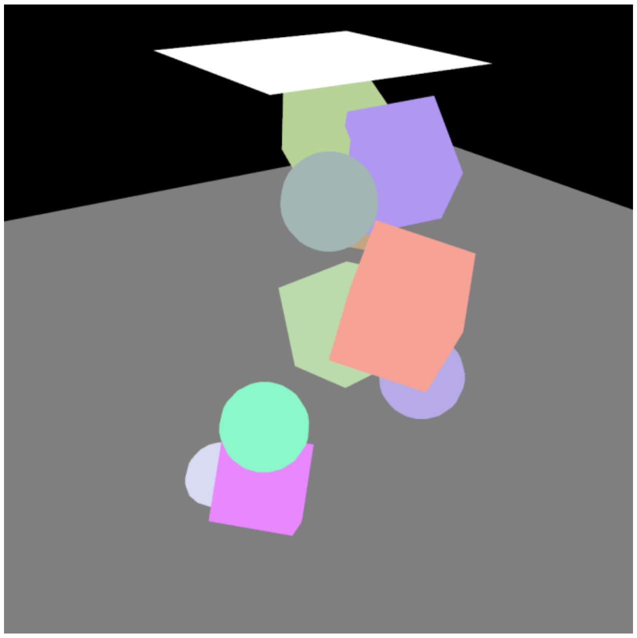
Octopus Demo

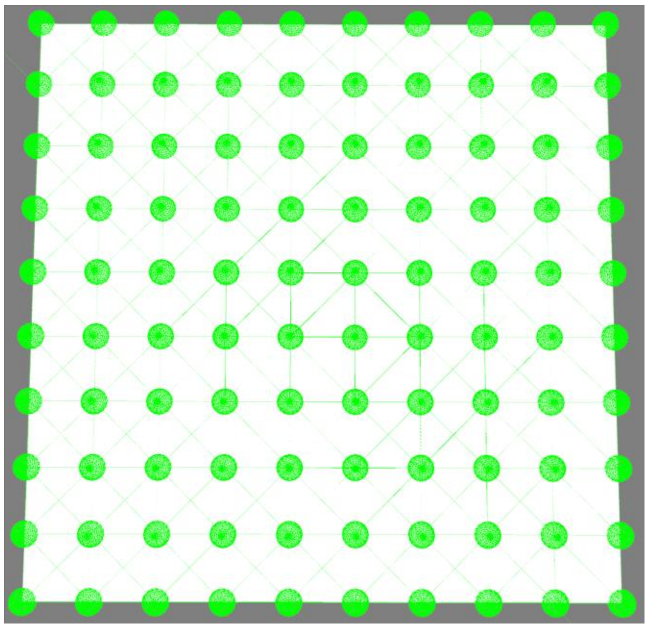
Multiple ropes attached to one object.



Cloth Demo

Many springs (*shown in right image*) combined to give cloth-like behaviour. (*Very unstable, looks more like melted cheese in gif…*)





Known Issues & Potential Improvements

Stability

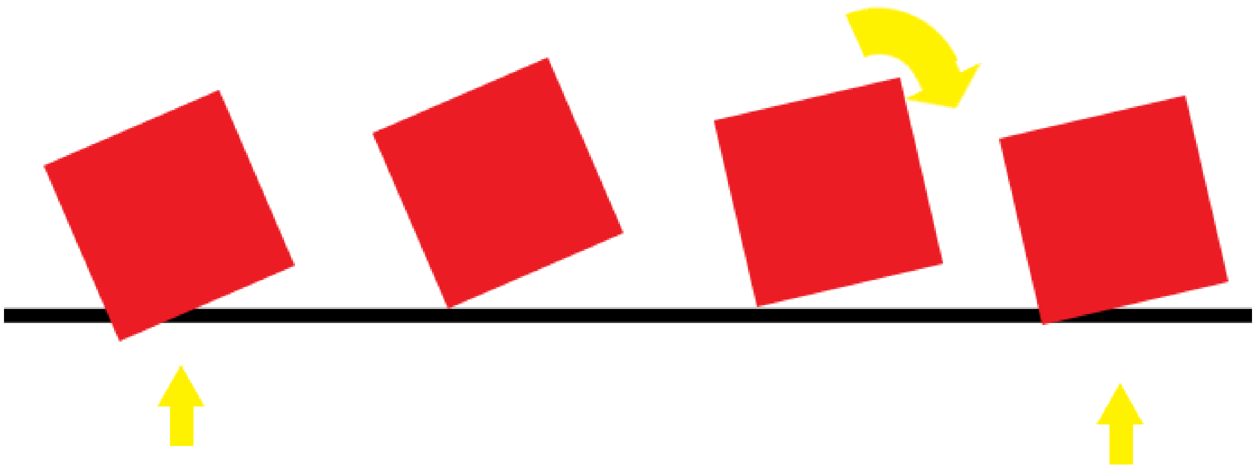
The engine is unstable (*especially springs and cloths*), this can be improved via a mixture of several methods:

- Using *[Runge-Kutta integration]* instead of the currently implemented *[Verlet integration]*, as it is much more accurate.
- *[Warm starting]*. Caching the impulses applied in the collision resolution phase and using it in the next physics step to converge towards the desired solution faster (*typically works best with objects that aren't moving much between frames*).

Collision Correction

To counteract objects sinking at rest, a force is applied away from the resting surface. The force applied does not consider rotations at this point and can cause the object to fall at an angle, rotating the object and repeating the process.

Linear positional correction, incurring additional unnecessary angular momentum. Yellow arrows show forces being applied during frame.



Performance

Performance of the physics engine is not ideal and can be further improved by reducing the number of collisions to detect – having a better broad phase algorithm will resolve this. (*An octree-based broad phase was in progress but remains incomplete currently*).

Visuals

As the engine uses the third dimension it can become very difficult to distinguish objects near each other, this is due to the lack of shadows or outlines on the objects; as this assessment was focused on physics this did not become a priority.

Third Party Libraries

Library	Description	Version	License	Source Code
GLFW	An open-source windowing library	v3.3.6	zlib/libpng license	GitHub
Glad	OpenGL loader library	v0.1.34	N/A	GitHub
glm	Header only mathematics library	v0.9.9.8	Modified MIT License	GitHub
ImGui	Immediate-mode Graphical User Interface library	v1.87	MIT License	GitHub
STB	Single-file libraries for C/C++	N/A	MIT License	GitHub
termcolor	Header only library to output colored messages to the console	v2.0.0	BSD 3-Clause License	GitHub

References

Fiedler, G. (2004, September 3). *Spring Physics*. Retrieved from Gaffer On Games: https://gafferongames.com/post/spring_physics/

Gaul, R. (2013, April 6). *How to Create a Custom 2D Physics Engine: The Basics and Impulse Resolution*. Retrieved from Envato Tuts+: https://gamedevelopment.tutsplus.com/tutorials/how-to-create-a-custom-2d-physics-engine-the-basics-and-impulse-resolution--gamedev-6331

Souto, N. (n.d.). *Video Game Physics Tutorial - Part I: An Introduction to Rigid Body Dynamics*. Retrieved from Toptal: https://www.toptal.com/game/video-game-physics-part-i-an-introduction-to-rigid-body-dynamics

Szauer, G. (2017). *Game Physics Cookbook.* Packt Publishing. Retrieved January 2022, from https://www.packtpub.com/product/game-physics-cookbook/9781787123663

Winterdev. (2020, August 1). Designing a Physics Engine in 5 minutes. Retrieved from https://www.youtube.com/watch?v=-_IspRG548E