CSB

Jack Diver

December 10, 2017

Abstract

Proposal for the Library assignment in the Programming Paradigms unit. BSc. Software Development for Animation, Games & Effects.

1 Introduction

I propose the creation of a library for the simulation of Cloth and Soft Body dynamics with a user imported mesh. The library would accept any industry recognized mesh format, and use the data to drive a Position Based Dynamics simulation. Simulation would be real-time and the library implmented in C++ using OpenGL, GLSL and Assimp.

2 Storage

Meshes would be stored within the program in one large pool of Points, no distinction is made between the different meshes once they have been loaded in as it is unnecessary for simulation. The edges of the mesh would be converted into simple distance constraints, however the user is able to request spring constraints with a provided stiffness. The pool of Points is will be implemented using a **std::vector**.

2.1 Points

A Point will only contain a position and velocity, as these are the only changing variables that are required by the simulation, (I'm trying to apply some Data-Oriented Design principles here), information used for drawing the points, such as the normals and shaders etc. will be stored separately, so that they aren't pulled in for the simulation calculations. Mass will likely be uniform for all points in a cloth so this can be stored separately to reduce the size of the Point Class. This may cause problems if the user wishes to have multiple objects with different masses, so the design may have to be revised to divide points into groups dependent on the mesh they were loaded from.

2.2 Constraints

Constraints will implement a common interface which includes a **project(bool, float, float)** function, where the three parameters represent equality, stiffness and inverse mass respectively. I chose to pass these as parameters, rather than store them in the constraints as they will likely be constant and uniform for all constraints. This function would be called to project the constraints on the estimated positions for the Points, until we obtain their new positions which satisfy those constraints. For efficiency reasons we will probably settle for a limited number of projections per time-step, as outlined in [Jak01], over multiple frames, even a small number of projections will give correct looking behavior. Example constraints could be distance, pin and bend. These constraints will most likely contain pointers, or references to Points which they directly manipulate, or even better, pointers directly to the positions.

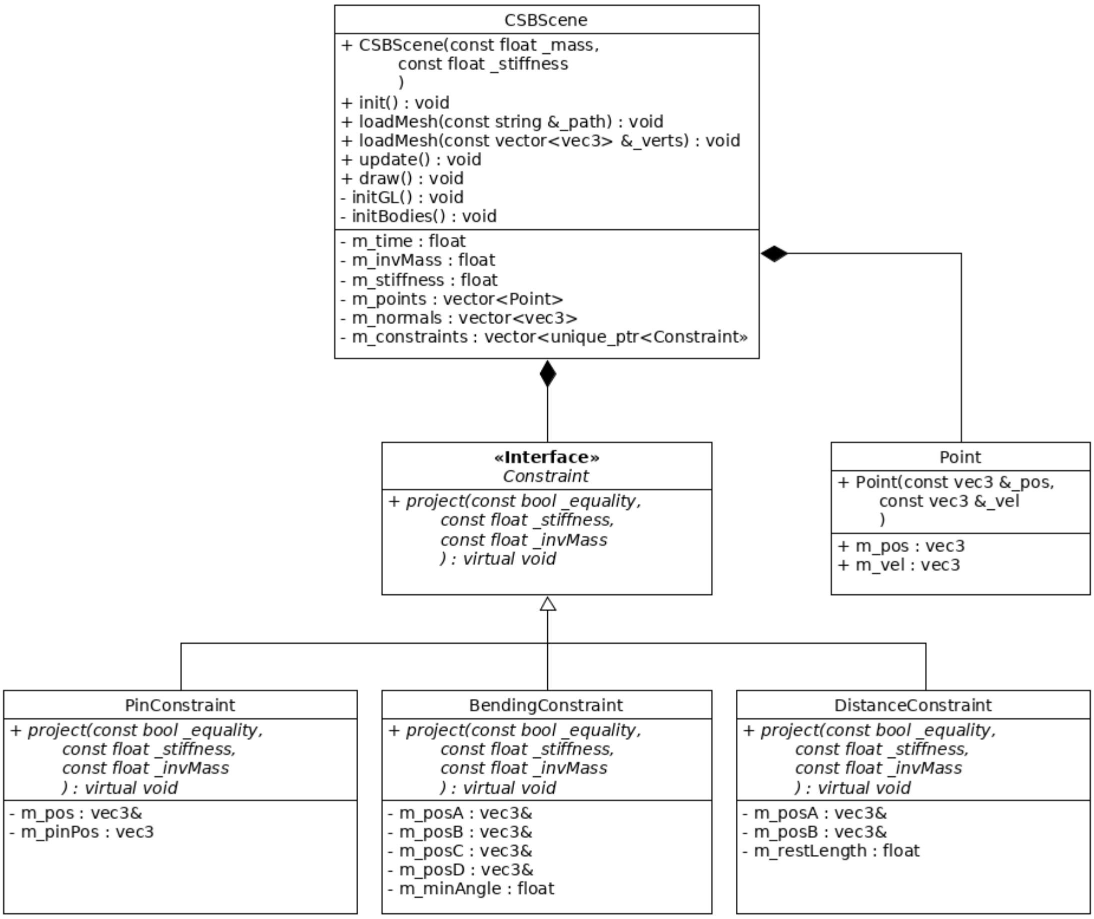
3 Algorithm

The simulation will use a Euler integration step to create estimated positions for the Points. Constraints will be used for direct manipulation of the Point positions, this will provide a more stable simulation and also allow for simple interactivity and direct Point positioning from the user, an example being click and drag for Points. The velocity of Points is stored explicitly rather than implicitly with a previous position to allow for simpler drag calculation. The constraint satisfaction step is in line with a Verlet integration, they will be projected over the estimated positions until they are satisfied, giving us the new Point Positions. Prior to calculating the estimates we can modify the Velocity to account for external forces. Collision will be solved by generating distance constraints after integration. The full method is described by [MHHR07].

4 Concerns

I am concerned that the use of a Euler integration step will slow down the simulation, as updating the position is dependent on the velocity, causing the position update to wait for the velocity. To solve this I may swap the Euler integration for a Verlet one. I am also concerned that my use of dynamic dispatch through the Constraint Class could cause a performance hit due to the sheer number of constraints that may be present, and the multiple iterations for projecting them.

5 Diagram



References

[Jak01] Thomas Jakobsen. Advanced character physics. In *Game Developers Conference*, volume 3, 2001.

[MHHR07] Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. Position based dynamics. *Journal of Visual Communication and Image Representation*, 18(2):109–118, 2007.