

Introduction

In this project I will be developing a 3D test environment in OpenGL to test two physics engines. In my literature review i will compare four engines. two closed source engines (Havok[1], PhysX[2]) and two open source engines (Bullet[3], ODE[4]).

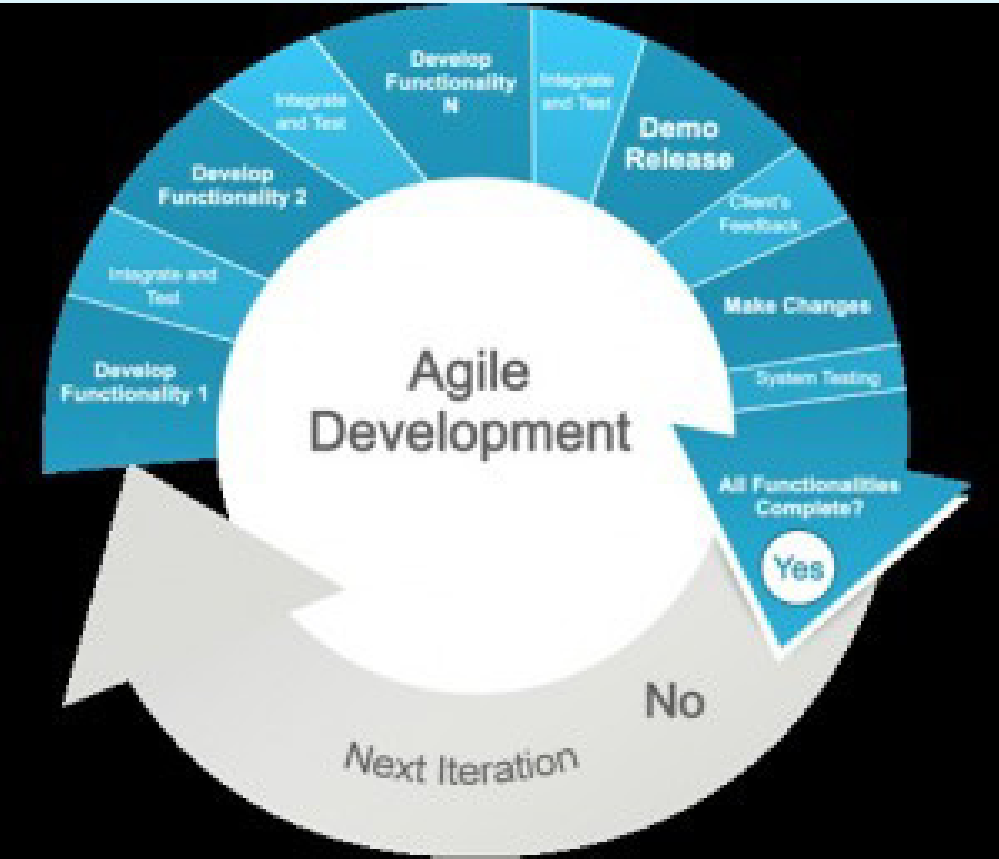
Objectives

I will be building a test environment using OpenGL and C++.  
that can easily swap between the different physics engines, with self generating log files that will write all the relevant data to files and automatically parse the data into a readable format.  
The objective is to compare two of the four physics engines running the same tasks using different engines, trying processes sequentially and in parallel

Methods

I will be using an agile development cycle so i can get a basic prototype built quickly and refine as needed.  
During the development lifecycle I will be doing test driven development throughout as efficiency and accuracy of data will be important.

Once the environment is built I will be testing the the engines using specific parameters(memory footprint, operation timing, FPS under load).



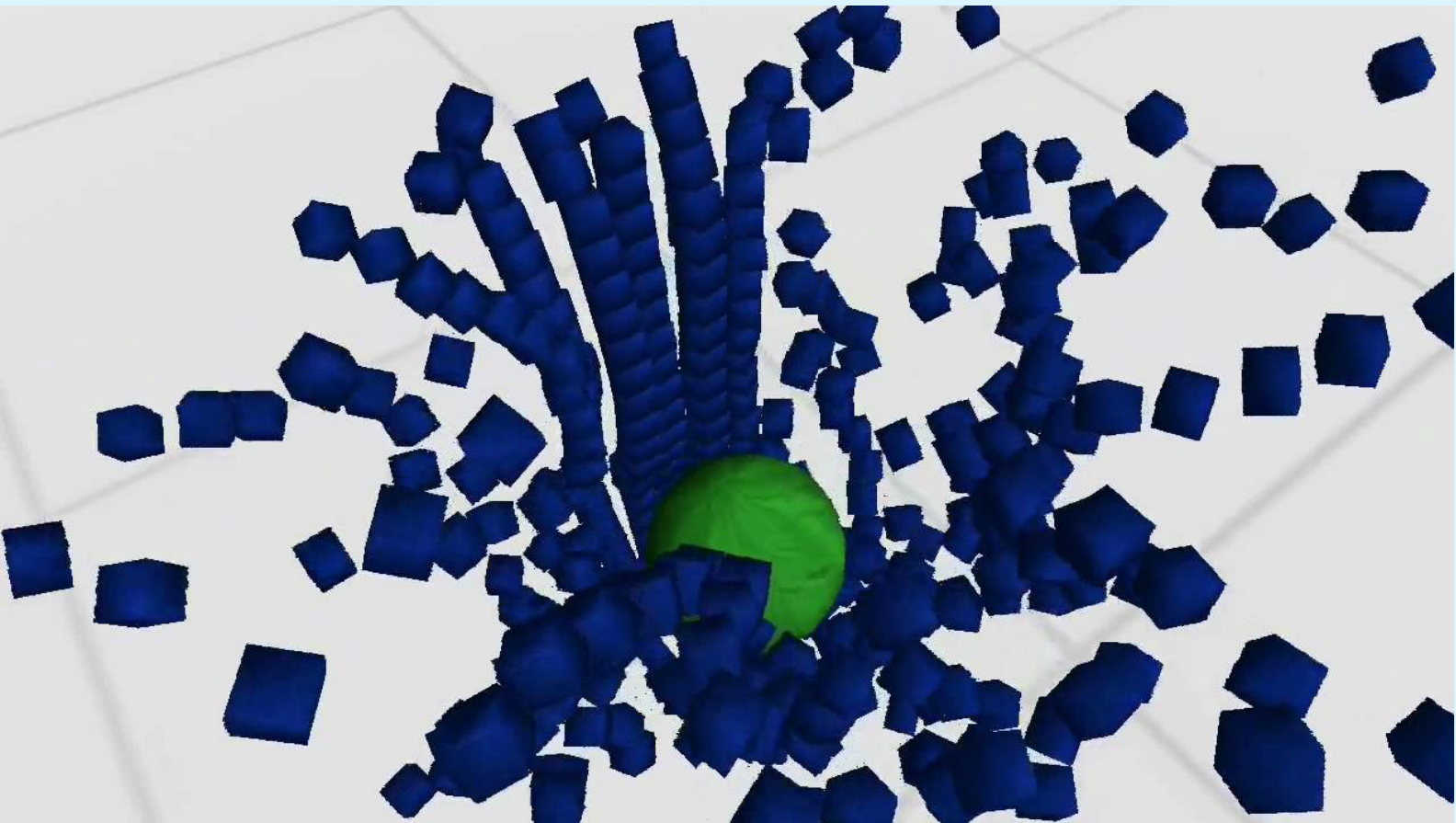
Requirements

Must have a UI

Must have all operations encompassed in the UI

Must be efficient in memory and hardware use so more resources can be used for the physics calculations

Must dump all necessary data into a human readable parsed format



Constraints

Constraints are my system, it needs a good processor, a good graphics card, a decent amount of RAM to be able to render enough primitives to be able to try and stress the engines

other constraints include an IDE, i need all of the physics engines I intend to test, I need a modern version of OpenGL, GLFW and GLEW for the graphics, and then i need to use proper benchmarking tools (CUDA etc) to be able to analyse my code.

Risk Analysis

The risks are that it might not work, or the environment itself might be too resource intensive to get decent performance with the physics engines.

I will mitigate these risks by doing proper testing of all the software, designing the software first using UML and by utilizing proper memory management.

Future Steps

Complete a literature review.

Design the software using UML.

Make the program.

References

[1] <http://www.havok.com/>  
[2] <http://www.geforce.com/hardware/technology/physx>  
[3] <http://bulletphysics.org/wordpress/>  
[4] <http://www.ode.org/>  
[5] <https://www.agsgov.com/what-we-do/software-development/agile-development-lifecycle-2/>  
[6] <https://i.ytimg.com/vi/ZrWqyl17DdQ/maxresdefault.jpg>