ANALYSIS PAPER

Introduction

The increase in computational power has made it possible to simulate the behaviour of particles in realistic situations. However there is still lack of speed in between the draw intervals. To rectify it we use various optimization techniques to make it much faster than it’s original time. Optimizing it not only makes it faster but also rectifies the number of function calls and memory intake required for each iteration.

Optimizing Techniques

Profilers

With the help of profilers we can identify the functions which utilize the most time

Compiler settings

We can tune up Compiler settings to tune our performance and stability of the system

Const

The **const** keyword allows you to specify whether or not a variable is modifiable. You can **use const** to prevent modifications to variables and **const** pointers and **const** references prevent changing the data pointed to (or referenced).

Convert Pointers to references

Pointers are references and references are pointers. The only difference is where they are stored. References are faster

De Morgan’s Law

De Morgan's Laws relate conjunctions and disjunctions of propositions through negation

Single Dimensional Arrays

Using Multi Dimensional Arrays (i,j) are slow since it take j times to Compute

Temprories

Temporaries are slow as the variable has to be assigned to temprory first

Encapsulation

Modifying set and get will provide us with more clean data

SIMD

SIMD is particularly applicable to common tasks such as adjusting the contrast in a digital image or adjusting the volume of digital audio.

STL

STL is reliable but slows down performance so it is better to use own own functions

Converting Double to Float

Double are useless and just wastes memory, Floats are much better to use