МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Компьютерные науки и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

Лабораторная работа №4
по курсу «Параллельная обработка данных»

Моделирование и визуализация системы N взаимодействующих тел с использованием технологий OpenGL и CUDA.

Выполнил: В. В. Бирюков
Группа: 8О-407Б
Преподаватель: А. Ю. Морозов

Москва, 2022

Условие

Цель работы: Использование GPU для моделирования и визуализации системы N взаимодействующих тел. Взаимодействие технологий CUDA и OpenGL: vbo + texture. Решение проблемы коллизий множества объектов. Создание простейшей «игры».

Программное и аппаратное обеспечение

Характеристики графического процессора:

- Наименование: NVIDIA GeForce GTX 1050 Ti
- Compute capability: 6.1
- Графическая память: 4294705152 Б
- Разделяемая память на блок: 49152 Б
- Количество регистров на блок: 65536
- Максимальное количество потоков на блок: (1024, 1024, 64)
- Максимальное количество блоков: (2147483647, 65535, 65535)
- Константная память: 65536 Б
- Количество мультипроцессоров: 6

Характеристики системы:

- Процессор: Intel(R) Core(TM) i5-6400 CPU @ 2.70GHz
- Память: 16 ГБ
- HDD: 2 ТБ

Программное обеспечение:

- ОС: Windows 10 Pro 22H2
- IDE: Visual Studio 2022 17.2.5
- Компилятор: nvcc 11.7.64

Метод решения

Для обработки коллизий между сферами, а также сфер с камерой, пулей и стенами используется электростатика — каждая сфера считается имеющей некоторый заряд, камера и пуля — больший заряд того же знака. Для препятствия прохождения сфер сквозь стены, каждая сфера дополнительно отталкивается от заряда такого же размера, спроецированного на каждую из стен.

Описание программы

Взаимодействие сфер рассчитывается самым примитивным образом — каждой с каждой. Для отрисовки карты напряженности так же необходимо для каждого пикселя учесть вклад каждой сферы. Это две самых вычислительно сложных части обновления всех объектов, поэтому они распараллеливаются. Во взаимодействии сфер параллельно рассчитывается результирующая скорость сферы, в карте напряженности — цвет пикселя.

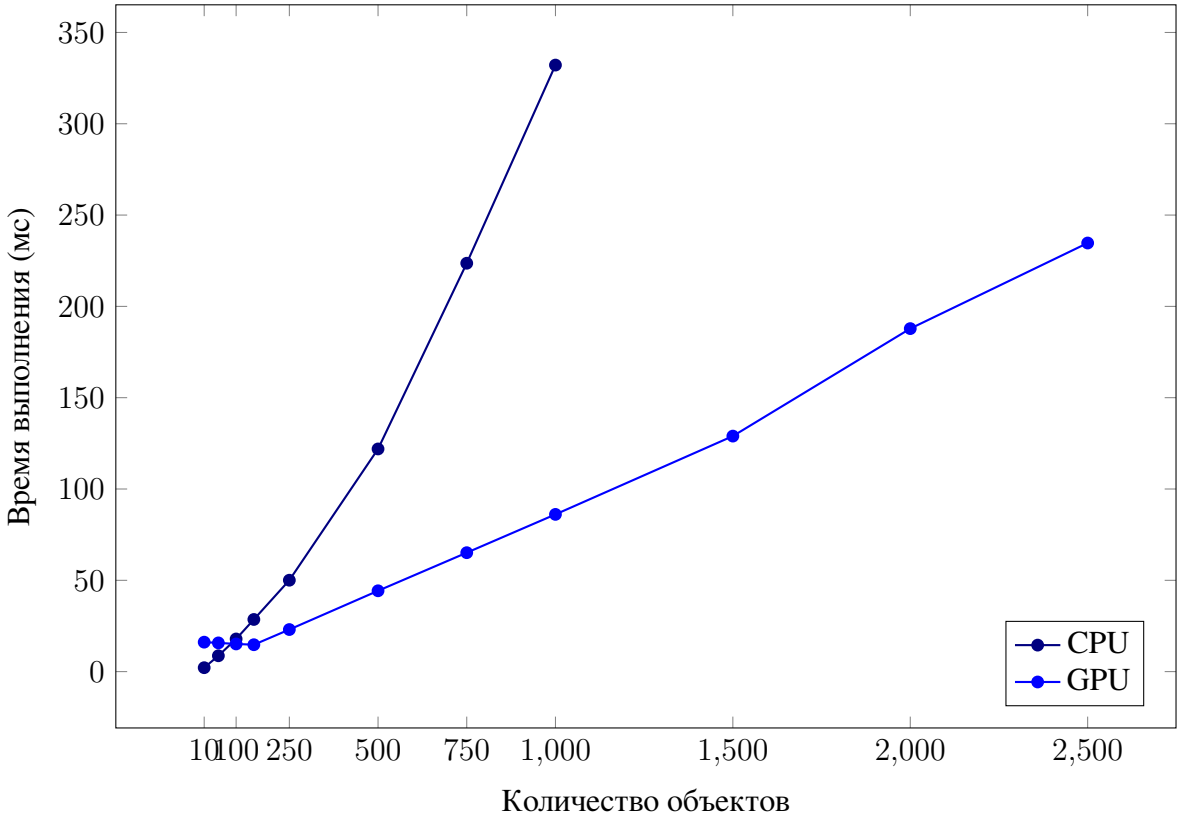
Результаты

Сравнение времени работы

В качестве замеров времени используем усредненное время выполнения функции `update` на первой 1000 кадров, при различном количестве сфер. Конфигурация ядра обработки сфер: необходимое количество блоков размером 256. Конфигурация ядра обработки пола фиксированная: 32×32, 32×8. Разрешение текстуры карты напряженности: 100×100.

Конфигурация	**Время выполнения, мс**					
Количество объектов	10	50	100	150	250	500
CPU	2.154	8.680	17.878	28.564	50.055	121.930
GPU	16.133	15.731	15.159	14.731	23.046	44.266
Количество объектов	750	1000	1500	2000	2500	
CPU	223.624	332.131				
GPU	65.126	86.098	128.968	187.829	234.696	

Из-за фиксированных конфигураций, время выполнения параллельной версии практически не отличается при маленьком количестве сфер, однако при 250 и более ее преимущество перед CPU версией более чем очевидно. При 1500 объектов производительность CPU версии падает до 2 fps, поэтому ее тестирование было остановлено на этом. GPU версия при таком количестве работает с 10 fps, что позволяет увеличивать его дальше, но при 2500 объектов их становится так много, что часть выдавливается за стенки куба.



Результаты работы

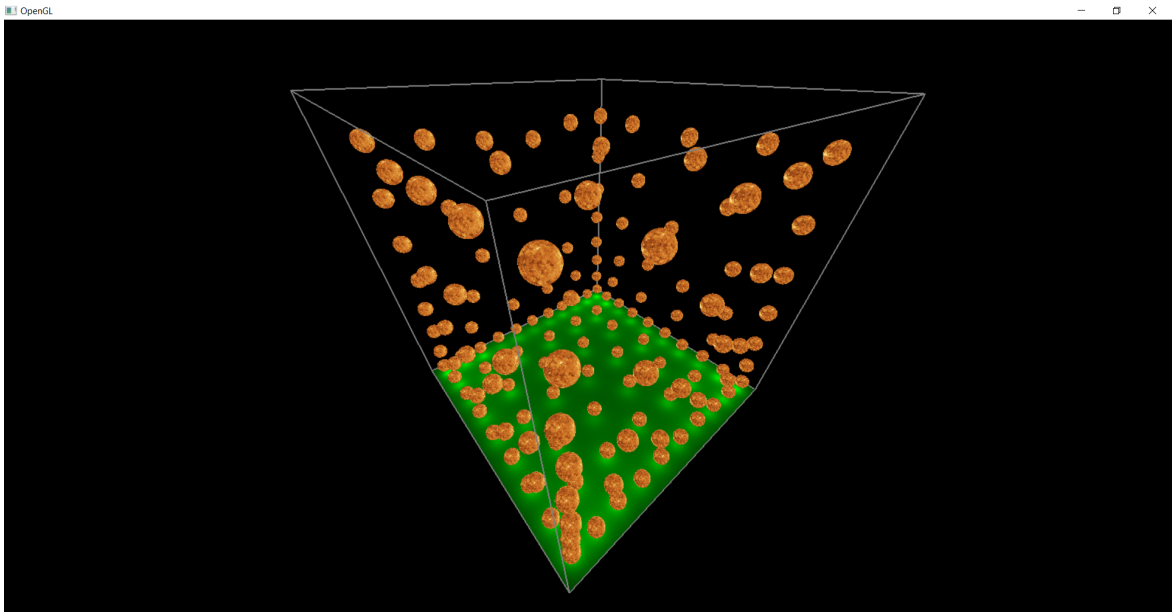


Рис. 1: Состояние равновесия

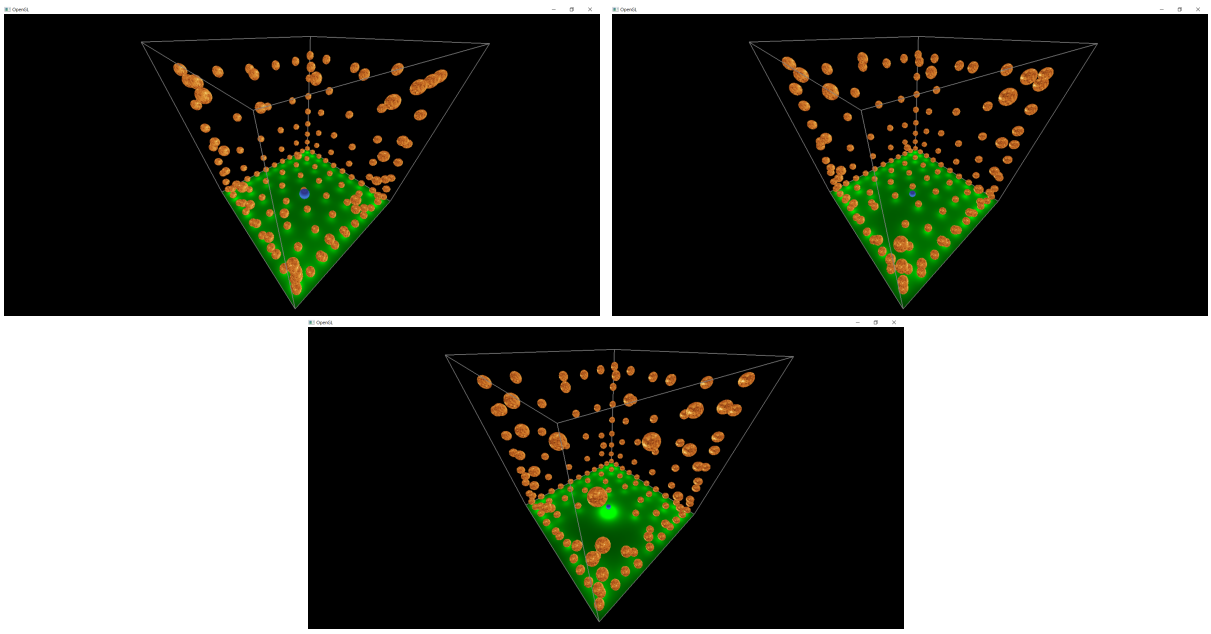


Рис. 2: Выведение из состояния равновесия снарядом

Выводы

В ходе выполнения лабораторной работы я познакомился с использованием CUDA в real-time компьютерной графике и ее взаимодействии с OpenGL.

Использование параллельных вычислений позволяет рассчитывать взаимодействие большого числа объектов без использования специальных алгоритмов и структур данных.