МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Компьютерные науки и прикладная математика»

Кафедра 806 «Вычислительная математика и программирование»

**Курсовая работа**

**по курсу «Параллельная обработка данных»**

***Обратная трассировка лучей (Ray Tracing) на GPU.***

Выполнил: Попов М. Р.

Группа: 8О-408Б

Преподаватели: К.Г. Крашенинников,

А.Ю. Морозов

Москва, 2023

**Условие**

1. **Цель работы:** Использование GPU для создание фотореалистической визуализации. Рендеринг полузеркальных и полупрозрачных правильных геометрических тел. Получение эффекта бесконечности. Создание анимации.
2. **Вариант 6:** тетраэдр, додекаэдр, икосаэдр.

**Программное и аппаратное обеспечение**

1. Графический процессор: Nvidia GeForce GT 545
   1. Количество потоковых процессоров: 144
   2. Частота ядра: 720 МГц
   3. Количество транзисторов: 1.170 млн
   4. Тех. процесс: 40 нм
   5. Энергопотребление: 70 Вт
2. OC: Ubuntu 16.04
3. Текстовый редактор: VS Code
4. Компилятор: nvcc

**Метод решения**

Для отрисовки кадров будем использовать треугольные полигоны, для отрисовки всех фигур и поверхности понадобится 62 полигона. В качестве алгоритма сглаживания используем SSAA. Рендеринг кадров и сглаживание выполним параллельно для разных пикселей посредством технологий CUDA.

**Описание программы**

Программа состоит из одного файла, в котором есть класс app, содержащий в себе все конфигурационные данные, необходимые для рендеринга, а также реализации алгоритмов сглаживания и рендеринга.

**Взаимодействие с программой**

Программа поддерживает следующие ключи:

* --default — запуск рендеринга с параметрами по умолчанию и вывод параметров в консоль;
* --cpu — запуск рендеринга на центральном процессоре;
* --gpu — запуск рендеринга на видеокарте, если также использовать –-cpu либо не использовать ни один из этих флагов, рендеринг выполнится на видеокарте.

При запуске программы без флага –default параметры для рендеринга нужно будет ввести в консоль в следующем порядке:

* количество кадров;
* путь к выходным изображениям;
* разрешение кадра (2 целых числа) и угол обзора в градусах (1 вещественное число);
* параметры движения камеры: , , , , , , , , , , , , , , , , , , , (20 вещественных чисел);
* параметры тетраэдра: координаты центра (3 вещественных числа), цвет (3 вещественных числа), радиус описанной окружности (1 вещественное число);
* параметры додекаэдра: координаты центра (3 вещественных числа), цвет (3 вещественных числа), радиус описанной окружности (1 вещественное число);
* параметры икосаэдра: координаты центра (3 вещественных числа), цвет (3 вещественных числа), радиус описанной окружности (1 вещественное число);
* параметры поверхности: координаты четырёх точек (12 вещественных чисел), цвет (3 вещественных числа);
* параметры источника света: положение (3 вещественных числа), цвет (3 вещественных числа), квадратный корень из количества лучей на один пиксель (1 вещественное число);

Пример входных данных:

1

%d.data

800 800 90

7 3 0 2 1 2 6 1 0 0

2 0 0 0.5 0.1 1 4 1 0 0

0 -2 0 1 0 0 1

0 0 0 0 1 0 1

0 2 0 0 0 1 1

-5 -5 -1 -5 5 -1 5 5 -1 5 -5 -1 1 1 1

10 0 15 0.294118 0.196078 0.0980392 4

**Результаты**

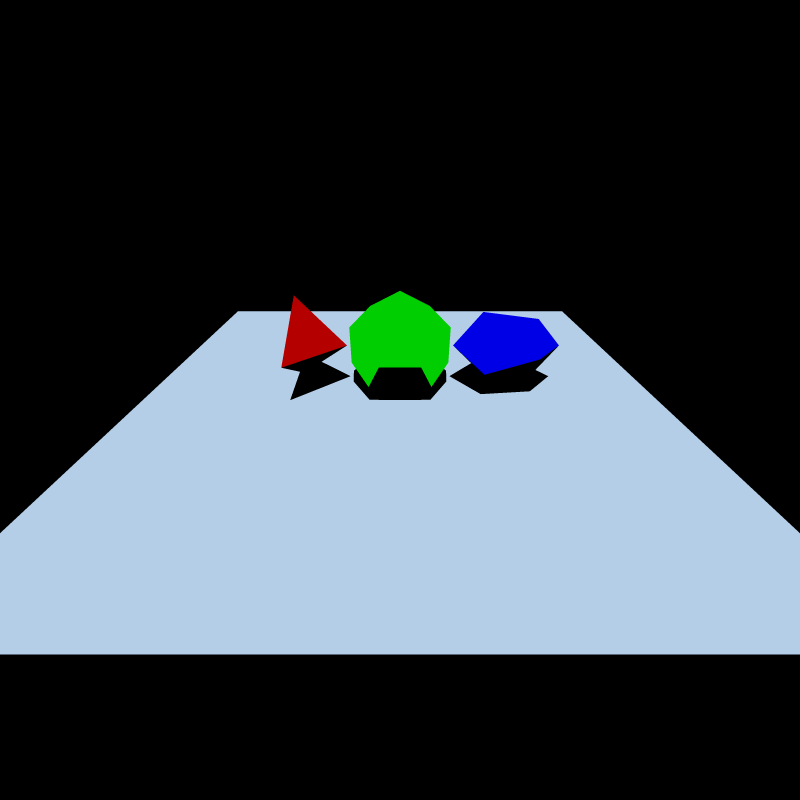
* + - 1. Зависимость времени выполнения программы от количества используемых потоков (рендеринг 50 кадров, разрешение 1600×1600):

|  |  |
| --- | --- |
| Потоки | Время (в сек) |
| 16×16×16×16 | 363 |
| 32×32×32×32 | 6 |
| 64×64×64×64 | 6 |

* + - 1. Сравнение программы на CUDA с 8×8×8×8 потоками и программы на CPU с одним потоком (рендеринг 25 кадров):

|  |  |  |
| --- | --- | --- |
| Размер кадра | Время на CUDA (в сек) | Время на CPU (в сек) |
| 400×400 | 12 | 270 |
| 800×800 | 48 | 1050 |
| 1600×1600 | 190 | 4200 |

1. Полученное изображение:

****

**Выводы**

Проделав лабораторную работу, я реализовал рендеринг изображений с помощью GPU, работал с полигонами, освещением и сглаживанием, сравнил скорость рендеринга на CPU и GPU и в очередной раз выяснил, что видеокарты в задачах, связанных с параллельными вычислениями, намного превосходят процессоры. В конечном итоге я получил изображение трёх объёмных фигур на поверхности со светом и тенями.