МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

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

Лабораторная работа №*2*
по курсу «Программирование графических процессоров»

Обработка изображений на GPU. Фильтры.

Выполнил: *И. П. Попов*
Группа: *8О-406Б*
Преподаватели: К.Г. Крашенинников,
А.Ю. Морозов

Москва, 2023

Условие

Цель работы. Научиться использовать GPU для обработки изображений.
Использование текстурной памяти и двухмерной сетки потоков.

Вариант 4. SSAA
Необходимо реализовать избыточную выборку сглаживания. Исходное изображение представляет собой “экранный буфер”, на выходе должно быть сглаженное изображение, полученное уменьшением исходного.
Входные данные. На первой строке задается путь к исходному изображению, на второй, путь к конечному изображению. На следующей строке, два числа wn и hn -- размеры нового изображения, гарантируется, что размеры исходного изображения соответственно кратны им. $w * h \leq 4 * 10^8$.

Программное и аппаратное обеспечение

NVIDIA GeForce GTX 1660 Super:
Compute capability: 7.5
Dedicated video memory: Typically 6 GB (may vary by manufacturer)
Shared memory per block: 49152 bytes
Register per block: 65536 bytes
Total constant memory: 65536 bytes
Max threads per multiprocessor: 2048
Max threads per block: 1024

CPU AMD Ryzen 3600X
Physical cores: 6
Threads: 12
Base clock speed: 3.8 GHz
Boost clock speed: 4.4 GHz
L1 cache: 384KB (per core)
L2 cache: 512KB (per core)
L3 cache: 32 MB (shared)
Chip lithography: 7 nm

16 Гб RAM
1 ТБ HDD
OS – Windows 11 ProWSL, IDE – VS Code, compiler - nvcc

Метод решения

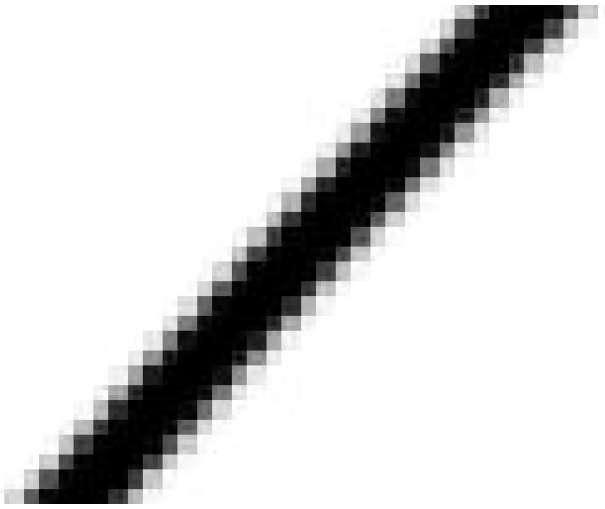
Метод решения, применяемый в программе, основан на избыточной выборке сглаживания (SSAA). Программа начинает с чтения исходного изображения, затем увеличивает его размер с использованием SSAA, чтобы улучшить качество и затем уменьшает до желаемых размеров, применяя усреднение значений пикселей внутри каждого нового пикселя. Этот метод позволяет уменьшить артефакты и улучшить общее визуальное качество изображения.

Описание программы

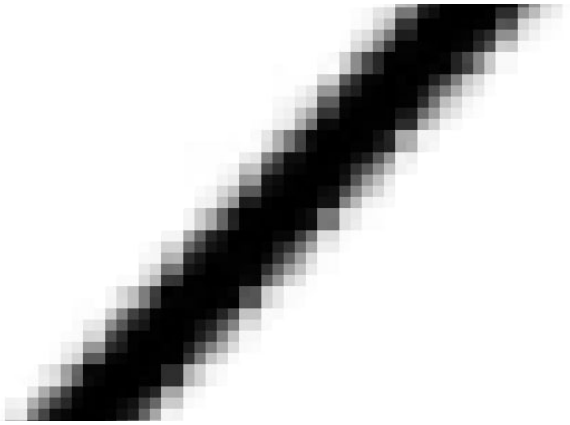
1. Чтение исходных данных
2. Чтение исходного изображения
3. Исходное изображение считывается с диска с использованием функции ReadData(). Это изображение хранится в формате uchar4
4. Настройка параметров для SSAA
   Вычисляются коэффициенты масштабирования wNorm и hNorm для уменьшения размера изображения.
5. Выделение памяти и копирование данных на GP
   Выделяется память на графическом процессоре для хранения данных исходного изображения (cudaArray).
6. Данные из исходного изображения копируются на графический процессор с использованием функции cudaMemcpy2DToArray().
7. Настройка текстурной памяти
   Задаются параметры текстурной памяти, такие как режим адресации, фильтрация и др.
   Функция cudaBindTextureToArray() используется для привязки текстурной переменной tex к выделенной памяти.
8. Выделение памяти на GPU для результата
9. Выполнение SSAA ядра
   Выполняется GPU-ядро kernel, которое использует избыточную выборку сглаживания для улучшения качества изображения. Ядро вычисляет среднее значение пикселей из исходного изображения внутри каждого нового пикселя после уменьшения изображения.
10. Копирование результата на CPU
11. Запись сглаженного изображения
    Сглаженное изображение записывается на диск с использованием функции WriteData().
12. Освобождение ресурсов

Примеры работы

Исходное изображение:



Обработанное изображение:



Результаты

Все измерения представлены в ms

	Размер изображения				
	Маленькие		Средние		Большие
	64*64	256*256	1024*1024	2048*2048	10000*10000
CPU	24.509314	248.263891	4518.887156	27266.701385	586179.892174
<<<(16,16), (16, 16)>>>	0.047368	0.062047	0.110812	0.286506	3.222785
<<<(32,32), (32, 32)>>>	0.059843	0.117245	0.254199	0.507781	3.019943
<<<(64,64), (32, 32)>>>	0.103135	0.506378	0.773615	0.863457	2.733272
<<<(128,128), (32, 32)>>>	1.117482	1.243194	1.369557	1.600949	4.158188

<<<(512,512), (32, 32)>>>	3.452968	3.790242	3.962729	4.294172	7.768719

Выводы

Программа эффективно использует текстурную память для улучшения производительности обработки изображений. Время выполнения на GPU остается стабильным для различных комбинаций размеров блоков и сеток, что свидетельствует о хорошей оптимизации и распределении ресурсов. Текстурная память позволяет ускорить доступ к данным изображения, особенно в случаях с множественным чтением изображения при избыточной выборке сглаживания (SSAA).