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: А. Л. Ядров
Группа: 8О-408Б
Преподаватель: А. Ю. Морозов,
К. Г. Крашенинников

Москва, 2023

Условие

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

Вариант: 7. Выделение контуров. Метод Собеля.

Программное и аппаратное обеспечение

Характеристики графического процессора:

- Наименование: NVIDIA GeForce GTX 1050
- Compute capability: 6.1
- Графическая память: 2047 Mb
- Разделяемая память на блок: 48 Kb
- Количество регистров на блок: 65536
- Максимальное количество потоков на блок: 1024
- Максимальное количество блоков: [1024, 1024, 64]
- Константная память: 64 Kb
- Количество мультипроцессоров: 5

Характеристики системы:

- Процессор: Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz
- Память: 8192 Mb

Программное обеспечение:

- ОС: Windows 11 (22H2)
- IDE: Microsoft Visual Studio 2022 (Comminity Edition)
- Компилятор: nvcc V12.2.140, MSVC V19.37.32824

Метод решения

Для выделения границ изображения необходимо осуществить на каждом его пикселе найти градиент якрости. Для этого применяются две операции свертки, по одной на компоненту вектора градиента. При этом свертка осуществляется не по исходным значениям цвета, а по величине яркости, вычисляемой по формуле $u = 0.299r + 0.587g + 0.114b$. По полученным компонентам находится модуль градиента, его значение необходимо ограничить максимальной величиной компоненты цвета – 255. В пиксель выходного изображения записывается модуль градиента во все цветовые каналы, альфа канал не изменяется. Сложность алгоритма — $O(w \cdot h)$.

Описание программы

Проект я разделил на следущие модули:

1. $common\ defines$ - содержит множество полезных дефайнов для более приятной работы с $CUDA$ и $C++$.

2. $common\ structures$ - содержит класс-обертку над $CUDA$ событиями и класс-обертку над $std::stringstream$, позволяющим конструировать строки ($sprintf$ с возможностями $C++$). Также сюда были помещены классы-обертки над текстурными объектами и ресурсами.

3. $operation\ system$ - содержит операции, связанные с системными вызовами (например, получение информации о количестве оперативной памяти). Реализован только для ОС $Windows$.

4. $kernel$ - непосредственно программа, выполняющия все операции, связанные с вводом, вычислениями и выводом.

Функция ядра осуществляет описанный выше алгоритм и принимает четыре аргумента: объект текстуры, указатель на массив входных данных и размеры изображения. Ядра свертки записаны в константную память, что позволяет кэшировать их и тем самым обеспечивать быстрый доступ к ним.

Также для замера времени была написана программа на языке программирования $Python$, состоящая из слеудющих модулей:

1. $build$ используется для компиляции $Microsoft\ Visual\ Studio$ проекта (файла с расширением $.sln$).

2. $tests\ generator$ используется для генерации тестов и ответов на них (ответы генерируются с помощью программы, написанной под CPU).

3. $cheker$ используется для проверки работоспособности программы. Для этого запускается раннее сгенерированный $.exe$ файл с входными данными из теста, далее вывод программмы проверяется с ответом.

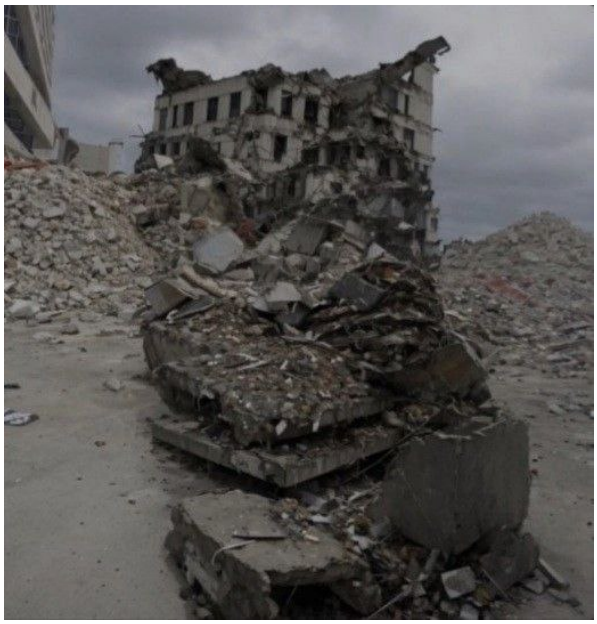
4. $benchmark$ используется для замера скорости работы программы. Для этого программа запускается 5 раз с входными тестовыми данными. Для каждого теста временем работы будет минимальное время работы из всех запусков данного теста.

5. $converter$ используется для конвертации изображения в бинарный вид и для получения изображения из бинарных данных.

6. $main$ используется в качестве обертке над описанными выше модулями для более комфортного запуска модулей.

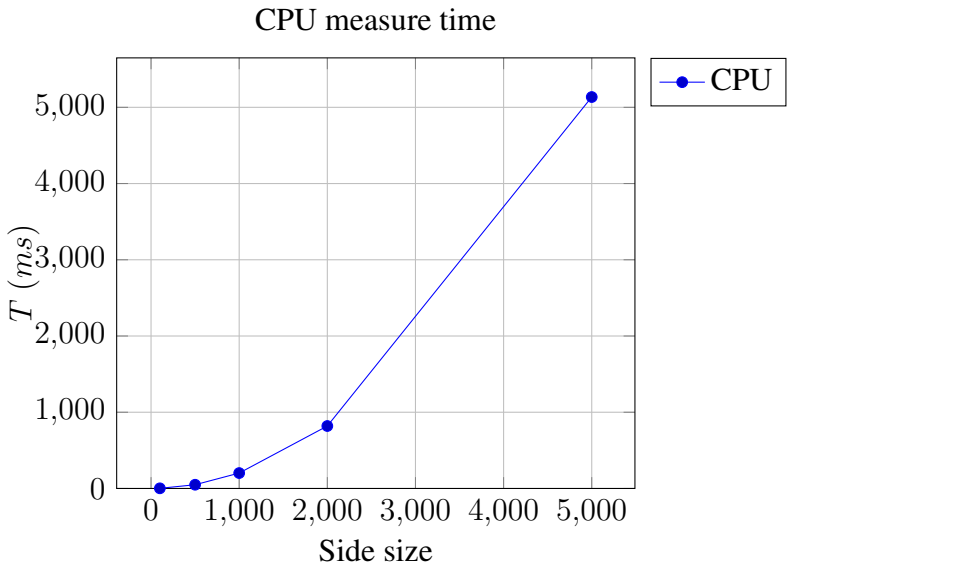
Результаты

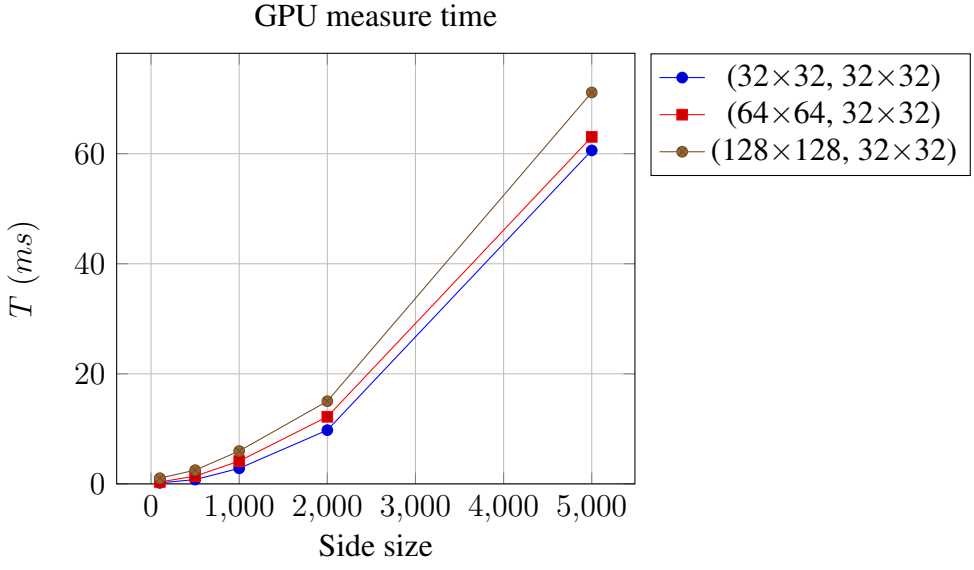
Примеры работы программы:



В качестве тестовых данных было взято квадратное изображение заданного размера. Пиксели генерировались с помощью генератора псевдослучайных чисел.

Для сравнения я написал программу, использующую только CPU, работающую в однопоточном режиме. Очевидно, что программа на GPU будет кратно быстрее, поэтому я разделил графики замеров работы.





Выводы

В ходе выполнения лабораторной работы я познакомился с обработкой изображений при помощи механизма свертки. Сверточные фильтры используются как непосредственно для обработки, так и, например, в компьютерном зрении.

Необходимо отметить, что текстурные объекты доступны только на GPU, чей параметр $compute\ capability\ \geqslant 3.0$. К счастью, моя видеокарта удовлетворяет этому условию, поэтому мне не пришлось менять подход к выполнению работы.

Список литературы

[1] Бьярне Страуструп. *Программирование. Принципы и практика с использованием C++, 2-е издание.* — Издательский дом «Вильямс», 2016. Перевод с английского: И. В. Красиков. — 1328 с. (ISBN 978-5-8459-1949-6 (рус.))

[2] *CUDA Runtime API.*
URL: `https://docs.nvidia.com/cuda/cuda-runtime-api/index.html` (дата обращения: 16.12.2013).