Московский Государственный Университет им. М.В. Ломоносова

Факультет Вычислительной Математики и Кибернетики

Кафедра Суперкомпьютеров и Квантовой Информатики



**Практикум на ЭВМ: 7 семестр.**

**Отчёт № 2.**

**Анализ улучшений программы на CUDA, реализующей свертку изображения**

|  |  |
| --- | --- |
|  | Работу выполнил  **Федоров В. В.** |
|  |  |

Москва 2021

# Постановка задачи и формат данных.

**Задача:** Реализовать улучшения программы с использованием CUDA, осуществляющей свертку изображения с тремя заранее выбранными ядрами (2 ядра 3х3 и 1 ядро 5х5), протестировать программу в двух режимах – на большом изображении размером минимум 2000х2000 и на множестве малых изображений размером порядка 300х300.

# Математическое описание

В общем виде свертка является операцией над произвольной матрицей размеров и другой матрицей размера , называемой ядром свертки. Выходом этой операции является матрица , вычисляемая по следующей формуле:

Для простоты элементы ядра нумеруем, начиная с . Для того, чтобы не выходить за границу входного массива, размер выходного массива получается равным . В случае изображения оно представляется в виде нескольких матриц, каждая из которых сворачивается по отдельности. Для тестирования программы были выбраны следующие 3 ядра:

Увеличение резкости (Sharpen):

Обнаружение границ (Edge detection):

Гауссово размытие (Gauss blur):

Примеры работы свертки с данными ядрами:

|  |  |  |  |
| --- | --- | --- | --- |
| Оригинал | Sharpen | Edge detection | Gauss blur |
|  |  |  |  |

# Описание программы

Программа состоит из следующих элементов:

* \_\_global\_\_ void conv(Pixel \*src, Pixel \*tgt, double \*ker, int ker\_pad, int width, int height); - функция, выполняемая CUDA-ядрами. Осуществляет свертку для одного пикселя выходного изображения.
* class CudaProcessor; - класс, хранящий в себе выделенные в оперативной памяти GPU массивы для входного и выходного изображения и для ядра, а также некоторые числовые параметры вроде размеров изображений и замеров по времени. Он имеет следующие методы:
  + CudaProcessor(int width, int height, double\* convker, int \_ker\_dim); - выделяет необходимую память на GPU.
  + void ProcessImage(Image& input, Image& output); - обрабатывает входное изображение и сохраняет результат в выходное.
  + ~CudaProcessor(); - освобождает выделенную память.
* int main(int argc, char \*argv[]); - основной цикл работы программы, открывает изображения, обрабатывает их и сохраняет, после чего выводит замер времени.

Программа написана с использованием CMake и библиотеки STB для открытия и сохранения изображений. После сборки из папки bin необходимо запустить программу при помощи команды ./main <sharpen/edge/gauss5> <large/small>

В случае аргумента large программа обрабатывает 1 изображение размером 5000х5000, в случае small – 1000 изображений размером 300x300.

Были реализованы следующие улучшения:

1. Использование отдельных функций для каждого фильтра вместо единой функции для всех.
2. Переиспользование динамической памяти, выделяемой библиотекой STB для открытия изображений. Вместо того, чтобы для каждого изображения выделять память заново и затем освобождать ее, программа сохраняет выделенную память и затем использует ее для новых изображений.
3. Для реализации параллельной обработки на девайсе копирования данных между процессором и девайсом каждое изображение обрабатывается в своем CUDA-потоке.
4. Была добавлена возможность загружать и обрабатывать N изображений за раз. Тестирование было проведено при N = 100.
5. После открытия изображения оно разделяется на 4 массива, в каждом из которых хранится отдельный канал, после чего каждый канал обрабатывается отдельно в своем CUDA-потоке.
6. Функции были переписаны с использованием разделяемой памяти: каждой нити соответствует пиксель из входного изображения и ячейка в разделяемой памяти, в которую она его копирует. После копирований нити синхронизируются и считают свертку на основе данных из разделяемой памяти.

**Тестирование и результаты**

Программа запускалась на системе Polus. Тестирование показало следующие результаты по времени:

**Wall-time работы программы, с**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ядро свертки | Режим работы | Оригинал | 1 | 2 | 3 | 4 | 5 | 6 |
| Sharpen | Large | 33.781229 | 31.100689 | 30.873911 | 31.644489 | 31.551375 | 32.005761 | 30.854136 |
| Small | 191.530448 | 180.008894 | 175.198975 | 178.124867 | 174.862174 | 175.701360 | 178.038064 |
| Edge | Large | 32.126672 | 31.040447 | 30.803197 | 31.903159 | 31.548928 | 32.905156 | 30.706696 |
| Small | 185.065419 | 181.572233 | 178.714117 | 176.291504 | 166.736041 | 167.609524 | 165.744487 |
| Gauss5 | Large | 31.684950 | 30.915566 | 30.819251 | 31.108766 | 31.192464 | 31.714750 | 30.515474 |
| Small | 177.057809 | 168.251438 | 167.364230 | 167.373238 | 166.464592 | 167.805969 | 166.419692 |

**Число обработанных пикселей / с**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ядро свертки | Режим работы | Оригинал | 1 | 2 | 3 | 4 | 5 | 6 |
| Sharpen | Large | 740056 | 803841 | 809745 | 743064 | 768017 | 781109 | 784827 |
| Small | 469899 | 499975 | 513702 | 505264 | 514691 | 512233 | 505510 |
| Edge | Large | 778170 | 805401 | 811604 | 759805 | 768074 | 759759 | 759724 |
| Small | 486315 | 495671 | 503598 | 510518 | 539775 | 536962 | 543004 |
| Gauss5 | Large | 789018 | 808654 | 811181 | 830323 | 828021 | 813941 | 847013 |
| Small | 508309 | 534914 | 537749 | 537720 | 540656 | 536334 | 540801 |

# Основные выводы.

В силу того, что большую часть времени программа тратит на открытие изображений с диска, ускорение работы GPU не дало значительных результатов. Использование индивидуальных ядровых функций для каждой свертки однозначно улучшило программу, действие остальных улучшений оказалось сомнительным.