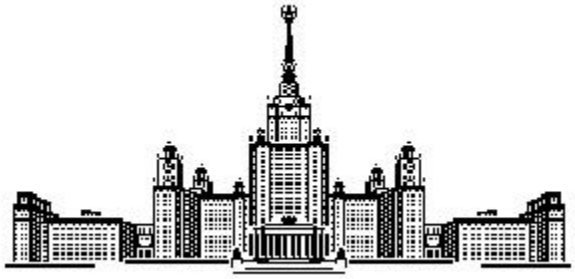
Московский Государственный Университет им. М.В. Ломоносова

Факультет Вычислительной Математики и Кибернетики

Кафедра Суперкомпьютеров и Квантовой Информатики



Практикум на ЭВМ
Отчёт № 1
Свёртка изображения с использованием CUDA

Работу выполнил

Сайбель Т. А.

Москва 2021

Постановка задачи

Реализовать программу с использованием CUDA, осуществляющую свёртку изображения с 3 фильтрами, и протестировать на 2 типах изображений: 2000x2000 и 300x300.

Описание алгоритма

Рассматривалось 3 фильтра:

Edge detection = $\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$

Gaussian blur = $\frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$

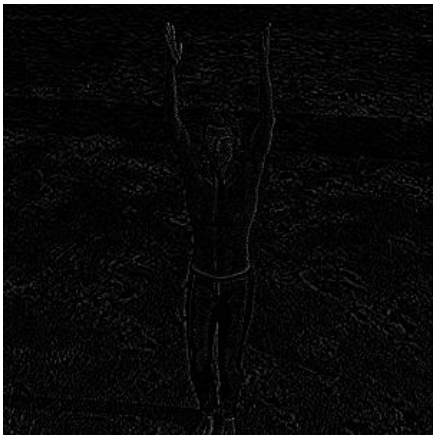
Sharpen = $\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$

Каждый элемент нового изображения вычислялся по следующей формуле:

$$\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix} * \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ y_{m1} & y_{m2} & \cdots & y_{mn} \end{bmatrix} = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} x_{(m-i)(n-j)}\, y_{(1+i)(1+j)}$$

Пример работы

Оригинал



Edge detection

Gaussian blur

Sharpen

Описание программы

Программы использует make и CImg(для чтения и записи изображения).
В качестве аргументов командной строки программа получает название фильтра (gaussian/edge/sharpen) и название файла с изображением (все изображения хранятся в image/, формат .jpg).
Программа выводит название фильтра и 2 времени выполнения: выполнение CUDA-ядер и выполнения CUDA-ядер + копирование данных, и сохраняет изображение в res/filename + filtername + “.jpg”.

Класс Solver: основной класс.
Класс Args: парсит командную строку.
Класс Reader: читает изображение
Класс Writer: сохраняет изображение
Класс Image: хранит изображение (Pixel*)

`void Solver::solve(int filter, const std::string &inFilename, const std::string &outFilename) - получает фильтр и название файла и сохраняет результат solve<filterName>().`

`Image Solver::solve<filterName>(Image) - Выделяет необходимую память на GPU и вызывает Kernel call.`

```
__global__
void applyFilter(Pixel *image, Pixel* filtere, const double * kernel, int kernelCenter, int width, int height) - осуществляет свёртку одного пикселя входного изображения
```

Результаты

Время работы CUDA-ядер

	Big image(3840x2160)	Small image(300x300)
Gaussian blur	9.36602	0.13296
Edge detection	3.68256	0.065536
Sharpen	3.66874	0.063488

Время работы CUDA-ядер и копирования данных

	Big image(3840x2160)	Small image(300x300)
Gaussian blur	19.2935	0.287904
Edge detection	13.534	0.225984
Sharpen	13.5458	0.221632