МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

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

Лабораторная работа №*3*
по курсу «Программирование графических процессоров»

Классификация и кластеризация изображений на GPU.

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

Москва, 2023

Условие

Цель работы. Научиться использовать GPU для классификации и кластеризации изображений. Использование константной памяти и одномерной сетки потоков.

Формат изображений соответствует формату описанному в лабораторной работе 2. Во всех вариантах, в результирующем изображении, на месте альфа-канала должен быть записан номер класса(кластера) к которому был отнесен соответствующий пиксель. Если пиксель можно отнести к нескольким классам, то выбирается класс с наименьшим номером.

Вариант 1. Метод максимального правдоподобия.

Входные данные. На первой строке задается путь к исходному изображению, на второй, путь к конечному изображению. На следующей строке, число nc -- количество классов. Далее идут nc строчек описывающих каждый класс. В начале j-ой строки задается число npj -- количество пикселей в выборке, за ним следуют npj пар чисел -- координаты пикселей выборки $nc \leq 32$, $np_j \leq 2^{19}$, $w * h \leq 4 * 10^8$.

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

Считываем входные данные. Используя эти данные для обучения, мы вычисляем на процессоре вектор средних значений, матрицы ковариаций, определитель матрицы ковариаций и их обратные матрицы. Затем мы копируем необходимые массивы в постоянную память. После этого мы вызываем ядро, в котором для каждого пикселя используем метод максимального правдоподобия (ММП) для определения предсказанного класса. В конечном итоге результат записывается в выходной файл, и выделенная память освобождается.

Описание программы

1. Определение константных массивов, которые содержат статистические данные для классификации пикселей (средние значения, ковариационные матрицы, их обратные матрицы и определители).
2. Описание функции classifyPixel, которая на GPU вычисляет класс пикселя с использованием метода максимального правдоподобия (ММП).
3. Определение ядра kernel, которое выполняет классификацию пикселей в параллель на GPU.
4. В функции main происходит основная логика программы:
    a. Считывание входных данных и параметров, включая изображение и статистические данные для классификации.
    b. Вычисление средних значений, ковариационных матриц, обратных матриц и определителей для каждого класса.
    c. Копирование данных в константную память GPU с использованием cudaMemcpyToSymbol.
    d. Выделение памяти на GPU и копирование изображения.
    e. Запуск ядра kernel для классификации пикселей.
    f. Копирование результата обратно на хост и запись в выходной файл.
    g. Освобождение памяти GPU и хоста.
5. Программа завершает свою работу, освобождая память и возвращая соответствующие коды возврата в случае успеха или неудачи.

Примеры работы

Исходное изображение:

Обработанное изображение:

Результаты

Все измерения представлены в ms

	Размер теста					
	O(10^3)	O(10^4)	O(10^5)	O(10^6)	O(10^7)	O(10^8)
<<< 1, 32 >>>	0.143244	19.867974	138.872361	488.238167	20936.173856	105839.246906
<<< 32, 32 >>>	0.032778	2.938261	13.769541	43.748226	2320.931594	10215.584767
<<< 256, 256 >>>	0.043276	1.671984	11.884379	34.935267	1489.725694	8502.346803
<<< 512, 512 >>>	0.109852	1.499821	11.734825	35.000581	1366.844509	8595.322086
<<< 1024, 512 >>>	0.189372	1.135646	11.856204	32.975741	1126.509472	8455.284173
CPU	223.456135	22376.584651	84212.316894	242336.865116	1546942.365186	5678441.253464

Выводы

Алгоритм, представленный в работе, предназначен для обучения с учителем и требует известных классов пикселей. Это ограничивает его практическую применимость, так как в реальных ситуациях мы часто должны предсказывать классы на новых и неизвестных данных, не имея предварительных меток.

Итак, хотя работа демонстрирует удачное использование GPU для обработки изображений, практическое применение данного алгоритма может быть ограничено из-за его требования заранее известных классов пикселей.