МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Компьютерные науки и прикладная математика»

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

Классификация и кластеризация изображений на GPU.

Выполнил: Филиппов Владимир Михайлович
Группа: М8О-410Б-22
Преподаватели: А.Ю. Морозов,
Е.Е. Заяц

Москва, *2025*

Условие

Кратко описывается задача:

1. Научиться использовать GPU для классификации и кластеризации изображений. Использование константной памяти и одномерной
2. Метод максимального правдоподобия

Программное и аппаратное обеспечение

GPU.

• Compute Capability: 8.9
• Графическая память (Global memory): 8 GB GDDR6 (8188 MiB — видно в nvidia-
smi)
• Shared memory per SM (мультипроцессор): до 100 KB (может конфигурироваться:
64 KB shared + 32 KB L1 или наоборот)
• Shared memory per block: до 48 KB
• Constant memory: 64 KB (аппаратно выделено)
• Регистры на потоковый мультипроцессор (SM): 65,536
• Максимум регистров на блок: 32K (зависит от конфигурации)
• Максимальное число потоков на блок: 1024
• Максимальное число блоков в сетке (grid): 2^31 - 1 по X, 2^16 - 1 по Y и Z
• Количество мультипроцессоров (SM): 24 (у RTX 4060)
• Макс. потоков на SM: 2048
• Итого потоков на GPU: 24 × 2048 = 49,152 одновременно резидентных

CPU.

• Архитектура. x86_64
• Модель. AMD Ryzen 5 3600 6-Core Processor
• Количество физических ядер. 6
• Потоков. 12 (2 потока на ядро)
• Частота (BogoMIPS). 7200.05
• Кэш L1 (данные/инструкции). 192 KiB / 192 KiB
• Кэш L2. 3 MiB
• Кэш L3. 16 MiB

RAM.

• Общий объём. 15 GiB
• Swap. 4 GiB (свободен)

SSD.• Объем. 1 TB
• Использовано. 156 GB

ПО.

• OS. Windows 11 + WSL 2.0.
• IDE. Visual studio code.
• NVCC. 11.5 (Build V11.5.119)

Метод решения

Каждой нити отводится некоторое количество пикселей. По формуле максимального правдоподобия вычисляем, насколько каждый из пикселей подходит тому или иному классу. Для этого предварительно предпосчитываем обратную ковариационную матрицу и логарифм определителя ковариационной матрицы.

Описание программы

Вся программа находится в файле main.cu. Присутствует одно ядро (classifyMLC), и несколько функций, которые вычисляются на CPU (computeMean, computeInvCov). Сначала, по заранее размеченным пикселям, для каждого класса вычисляются вектор средних значений, ковариационная матрица, её обратная и логарифм детерминанта – это этап обучения. Далее для каждого пикселя изображения на GPU мы запускаем функцию classifyMLC, которая уже непосредственно относит каждый пиксель к тому или ному классу, используя для этого метод максимального правдоподобия, который основан на расстоянии Махаланобиса.

Результаты

По традиции, я составил тепловую карту, которая представлена на рисунке 1, скорости исполнения ядра на GPU. Замеры были произведены на изображении 3000x3000 пикселей и при классификации на два класса, при этом все результаты были усреднены 20тью запусками. По стандарту, при увеличении количества блоков и потоков, скорость исполнения увеличивается, но при этом видим новый интересный эффект: если количество потоков кратно 64, то производительность падает вне зависимости от количества блоков, а вот максимальная производительность наблюдается при количестве потоков кратном 96. Видимо это связано с тем, что при количестве потоков в блоке загружаются все 24 SM, которые доступны на моей видеокарте.

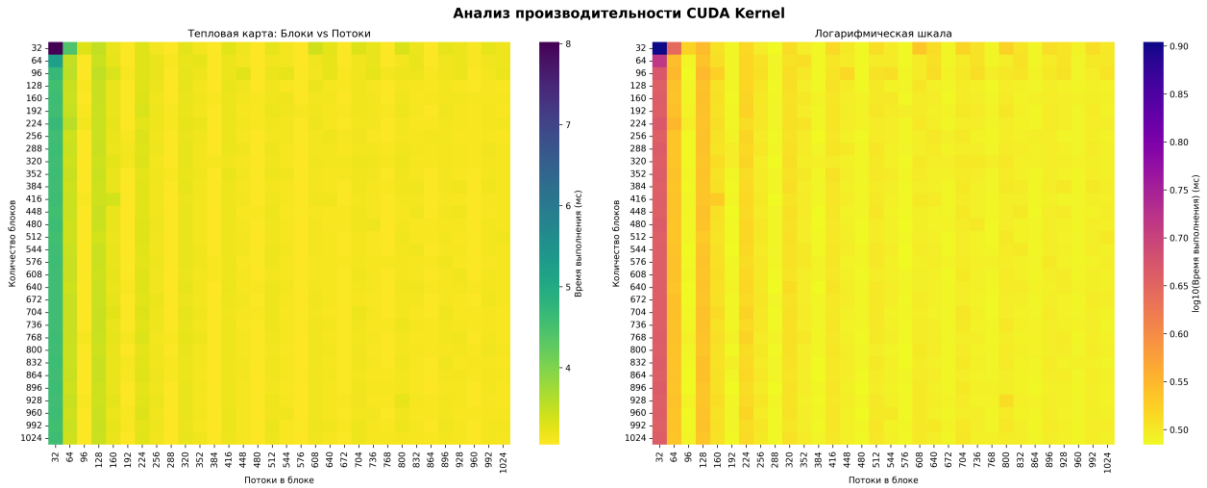

Рисунок 1. Heatmap производительности алгоритма относительно количества блоков и потоков.

	CPU (мс)	GPU(мс)
9000x9000	1959.4	27.662474
6000x6000	874.25	12.337056
3000x3000	217.6	3.046581

Таблица 1. Сравнительная характеристика времени исполнения на GPU и CPU.

Выводы

Данный алгоритм используется, например, в дистанционном зондировании и обработке изображений, когда необходимо автоматически классифицировать каждый пиксель по его спектральным характеристикам на один из заранее определённых классов (например, вода, растительность, почва, застройка). С его помощью изображение разбивается на тематические регионы, что позволяет анализировать распределение объектов или материалов на снимке.

В реализации алгоритма MLC используется метод максимального правдоподобия: для каждого пикселя вычисляется вероятность принадлежности к каждому классу на основе многомерного нормального распределения в цветовом (или спектральном) пространстве. Пиксель присваивается классу с максимальной вероятностью. Несмотря на простоту идеи, вычисление для всех пикселей является ресурсоёмким, так как для каждого пикселя необходимо просчитать квадратичную форму с обратной ковариационной матрицей каждого класса, то есть сложность порядка O(N * C), где N – число пикселей, а C – число классов.