ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

Федеральное государственное образовательное учреждение высшего профессионального образования ордена Трудового Красного Знамени

«МОСКОВСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ СВЯЗИ И ИНФОРМАТИКИ»

МКиИТ

**ЛАБОРАТОРНАЯ РАБОТА №9**

По дисциплине «Распределенные и параллельные вычисления»

На тему **«Разностная схема»**

Выполнил студент 2 курса группы МБД2032:

Трояновский Семён Владиславович

Москва

2021

**Задача лабораторной работы:**

Цель данной лабораторной – реализовать блочный алгоритм свертки изображения, используя как разделяемую, так и константную память.

**Ход выполнения:**

Для начала реализуем алгоритм расчётов:

1. **Инструменты**

В данной работе принято решение использовать уже знакомую мне библиотеку numba, а также cupy для упрощения работы с векторами и GPU.

1. **Полный код**

Полный код программы в моем случае выглядит так:

from numba import stencil, cuda

import numba.cuda

import cupy, math

@numba.cuda.jit

def smooth\_gpu(x, out):

i, j = cuda.grid(2)

n, m = x.shape

if 1 <= i < n - 1 and 1 <= j < m - 1:

out[i, j] = (x[i - 1, j - 1] + x[i - 1, j] + x[i - 1, j + 1] +

x[i, j - 1] + x[i, j] + x[i, j + 1] +

x[i + 1, j - 1] + x[i + 1, j] + x[i + 1, j + 1]) // 9

x\_gpu = cupy.ones((10000, 10000), dtype='int8')

out\_gpu = cupy.zeros((10000, 10000), dtype='int8')

# I copied the four lines below from the Numba docs

threadsperblock = (16, 16)

blockspergrid\_x = math.ceil(x\_gpu.shape[0] / threadsperblock[0])

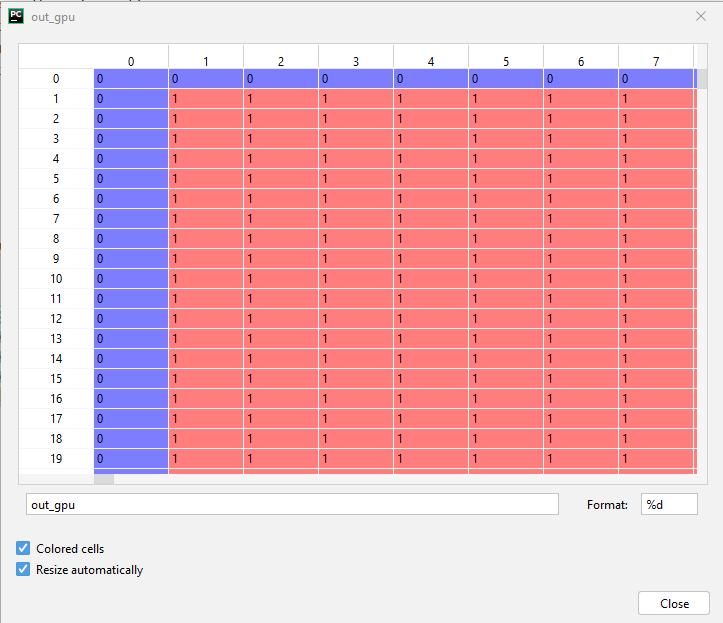
blockspergrid\_y = math.ceil(x\_gpu.shape[1] / threadsperblock[1])

blockspergrid = (blockspergrid\_x, blockspergrid\_y)

smooth\_gpu[blockspergrid, threadsperblock](x\_gpu, out\_gpu)

1. **Результат**

Финальная матрица выглядит так:

****

1. **Ответы на вопросы**

(1) Сколько чтений из глобальной памяти выполнит программа?

*Данный код выполнит BLOCK\_SIZE \* BLOCK\_SIZE \* ceil(size\_x/TILE\_SIZE) \* ceil(size\_y/tile\_size) чтений.*

(2) Сколько чтений из разделяемой памяти выполнит ваша программа?

*Каждый поток сделает 7 чтений, всего BLOCK\_SIZE \* BLOCK\_SIZE \* ceil(size\_x/TILE\_SIZE) \* ceil(size\_y/TILE\_SIZE)потоков.*

(3) Как бы вы использовали готовый код, выполняющий свертку, для расчета данной разностной схемы?

*Для данной схемы потребуется свертка с ядром 3x3, где центральный элемент равен 6, смежные к нему – 1. Угловые элементы равны 0.*

(4) Ваш код, рассчитывающий разностную схему, делает столько же, меньше или больше обращений к памяти,

чем эквивалентный код расчета свертки 3x3?

*Данный код выполнит меньшее количество обращений. Эквивалентный код, проводящий вычисления через свертку, также будет загружать значения, соответствующие нулевым угловым элементам маски.*

1. **Вывод**

В данной работе я реализовал расчёт разностной схемы с использованием CUDA API.