Лабораторна робота №5

**Тема**: CUDA: Робота із зображеннями.

**Мета**: Навчитись працювати із CUDA бібліотеками (Python) для обробки зображень.

1. Теоретичні відомості

Обробка великих зображень за допомогою центрального процесора може зайняти тривалий час. Для того, щоб прискорити обробку, можна використовувати технологію NVIDIA CUDA. Для обробки зображень за допомогою CUDA існує досить багато бібліотек. Загалом бібліотеки використовуються для того, щоб не писати багато алгоритмів з нуля, що може зайняти багато часу. Розглянемо найпопулярніші бібліотеки для обробки зображень, які можна застосовувати разом із технологією CUDA.

*Scikit-image.* Це бібліотека з різноманітними алгоритмами для обробки зображень. З коробки дана бібліотека не підтримує роботу з GPU, але це можна вирішити використанням наступних бібліотек разом із цією. Посилання на офіційний сайт - <https://scikit-image.org/>

*сuCIM.* Це бібліотека програмного забезпечення для комп’ютерного зору та обробки зображень із відкритим кодом для багатовимірних зображень, які використовуються в біомедичних, геопросторових, матеріалознавчих і біологічних науках, а також у випадках використання дистанційного зондування. Ця бібліотека пропонує розширені можливості обробки зображень для великих і n-вимірних файлів із зображеннями тегів (TIFF); прискорену продуктивність завдяки обробці зображень на основі графічного процесора (GPU) і примітивів комп’ютерного зору; простий інтерфейс. Посилання на репозиторій - <https://github.com/rapidsai/cucim>

*CuPy*. Дана бібліотека практично є аналогом іншої бібліотеки Numpy, проте з підтримкою GPU. NumPy — це бібліотека для мови програмування Python, яка додає підтримку великих багатовимірних масивів і матриць разом із великою колекцією математичних функцій високого рівня для роботи з цими масивами. Посилання на репозиторій - <https://github.com/cupy/cupy/>

Також найпопулярнішою бібліотекою комп’ютерного зору є OpenCV. Дана бібліотека має реалізацію практично під кожну популярну мову програмування (C, Java, Python) - <https://opencv.org/>. Дана бібліотека має «найбагатший» функціонал із всіх вищенаведений. Проте в неї є один мінус. Вона не підтримує GPU модуль із коробки. Це означає, що дану бібліотеку потрібно вручну скомпілювати із включеним GPU модулем. Оскільки це досить кропітка робота (<https://thinkinfi.com/install-opencv-gpu-with-cuda-for-windows-10/>), то ми не будемо розглядати роботу із даною бібліотекою у цій лабораторній роботі.

2. Приклад програми

Нижче наведено приклад програми, яка відкриває зображення, записує його а пам’ять GPU, зчитує назад та відображає на екрані.

Для початку необхідно встановити потрібні бібліотеки.

pip install cucim

pip install scipy scikit-image cupy-cuda100

Далі імпортуємо їх:

import numpy as np

import cupy as cp

import cucim

from skimage.io import imread, imshow

Для того щоб зчитати зображення використовуємо наступний код:

image **=** imread('https://idr.openmicroscopy.org/webclient/render\_image\_download/9844418/?format=tif')

imshow(image)

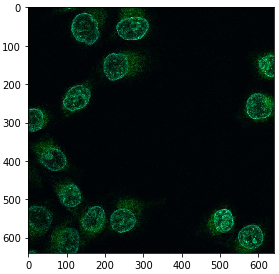


Рисунок 1. Вхідне зображення

Щоб обробити зображення за допомогою CUDA на GPU, нам потрібно його конвертувати. Під капотом цього перетворення дані зображення надсилаються з комп’ютерної оперативної пам’яті (RAM) у пам’ять GPU.

image\_gpu **=** cp.asarray(image)

Далі спробуємо відобразити лише один канал цього зображення. Операція ідентична до тої, якби ми працювали з numpy. Показ зображення за допомогою scikit-image imshow не працює, оскільки зображення CUDA недоступне в пам’яті. Щоб отримати його назад із пам’яті GPU, нам потрібно перетворити його на масив numpy.

single\_channel\_gpu **=** image\_gpu[:,:,1]

single\_channel **=** np.asarray(single\_channel\_gpu)

imshow(single\_channel)

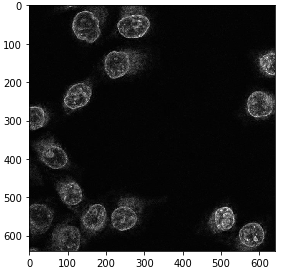


Рисунок 2. Одноканальне зображення з пам’яті GPU.

2. Хід роботи.

Використовуючи приклади коду вище написати програму відповідно до варіанту:

1. Гаусовий фільтр та порогова сегментація.
2. Медіанний фільтр та визначення порогу розмиття зображення (blur\_effect).
3. Операція Dilate та Erode.

В якості вхідного зображення можна використовувати зображення із прикладу вище.

3. Структура звіту лабораторної роботи.

* Титульна сторінка.
* Тема та мета роботи.
* Короткі теоретичні відомості.
* Лістинг коду та екрани із результатами виконання програми.
* Висновки.

4. Контрольні запитання

1. Які бібліотеки для опрацювання зображень ви знаєте?

2. Що таке бібліотека?

3. Чому опрацьовувати великі зображення на GPU швидше ніж на CPU?

Література:

1. <https://scikit-image.org/docs/stable/api/api.html>
2. <https://docs.cupy.dev/en/stable/user_guide/basic.html>
3. <https://developer.nvidia.com/blog/cucim-rapid-n-dimensional-image-processing-and-i-o-on-gpus/>