GPU Computing – Project: An Implementation of Image Processing Library

ANKITA DAGAR, MT18068; AKANKSHA MALHOTRA, MT18062

The task for the project is to build Image Processing Library on the GPU. We have developed a library which provides basic primitive functions on images using parallel programming which can add acceleration to imaging applications. We have also done a comparison with corresponding naive CPU code for performance.

Concepts: • **Image Primitive Functions** → Digital Image Processing; • **GPU** → Parallel Programming.

1 LITERATURE SURVEY

Digital Image Processing's motive is to improve the quality of an image. Images are used in number of application domains starting from medical, industrial to defense. Digital Image Processing requires processing of large amounts of data which increases the computational power and processing time to a great extent. The image operations are generally highly iterative and often use identical operations on all pixels. A simple operation on grey image with 1000 x 1000 pixels will require millions of operation. However, parallel processing of the image data can significantly reduce the processing time. The programmable Graphic Processor Unit or GPU is useful for such extremely parallel data workloads, where similar calculations are executed on quantities of data that are arranged in a regular grid-like fashion, so it is one of ideal solutions of large size images. Since image operations are iterative in nature, which makes it ideal application for parallel computing. GPUs has matured into a multithreaded, highly parallel, manycore processor with massive computational power and very high memory bandwidth. The highest performing parallel computing processor , Quadro GV100 has 14.8 teraflops of single precision. The potential application of parallelism to reduce the computation cost and increase the performance of various image processing algorithms is the motivation behind building a GPU -accelerated image processing library.

2 ANALYSIS OF ALGORITHM

In sequential implementation, the time taken by each operation is O(height * width * imageChannels). In our parallel implementation we have created Tile_width*Tile_width number of threads. And assigned constant work to each thread. So, the total time taken by each operation will remain constant.

3 PARALLELIZATION STRATEGY AND IMPLEMENTATION NOTES

We have implemented some operations in the library and have compared them with their CPU versions. We have made a header file with the parallel version of the mentioned operations. All logical, geometry and point operations are working on RGB as well as grayscale image of any size. The local operations work on images with noise. The operations implemented are as follows (for now):

LOGICAL:

- Not- Inverts the bits of each pixel.
- And- Bitwise AND operation between two images.
- Or- Bitwise OR operation between two images.
- Xor- Logical bitwise XOR operation between two images.

GEOMETRY:

- Flip Horizontal- Mirrors an image about a horizontal axis.
- Flip Vertical- Mirrors an image about a vertical axis.
- Rotate Anti-Clockwise- Rotates an image in anti-clockwise direction.
- Rotate Clockwise- Rotates an image in clockwise direction.
- Crop- Crops the desired portion of an image.

POINT OPERATIONS:

- Invert- Inverting the pixel values in the image similar to be found in a film negative.
- Brightness- To change the brightness of the image by passing positive value to increase brightness and a negative value to decrease brightness.

LOCAL OPERATIONS:

LINEAR:

- Mean- Filters an image using a mean filter (new value of a pixel is the arithmetic mean of the pixel values in its neighborhood).
- Gaussian- Filters an image using a gaussian filter (new value of a pixel by calculating the mean of the pixel values in its neighborhood, weighted by a gaussian distribution.).

4 RESULTS

The speed up vs Image Size table is shown in Table.1.

	64x64	128x128	256x256	512x512	1024x1024	2048x2048
Inverse	0.824282389	0.681719023	20.23613722	21.49470899	23.20746036	24.32946621
Brightness	5.319148936	14.38601322	25.68667345	35.40152016	34.21762179	33.50563556
Flip Vertical	3.460207612	9.876179245	17.63416578	23.26872822	24.3927309	26.07579519
Flip Horizontal	3.242924528	9.588068182	17.12372449	22.6148558	24.1774363	25.71064815
Anticlockwise rotation	4.024621212	10.94543147	22.73061105	41.43518519	50.5056165	82.90256715
Clockwise Rotation	3.876879699	11.13782051	21.90420561	28.62323753	51.73137461	104.7224896
And	4.765070922	11.68536325	19.33019301	24.29950639	26.51844246	28.18681862
OR	4.765070922	12.09900442	19.09090909	24.00947459	26.5144041	27.55291329
XOR	4.952830189	11.76948052	18.48088752	23.92074742	26.52906104	28.12479436
Not	3.578244275	9.876943005	15.3385947	19.36848958	21.76507538	23.68156899
Crop	0.119731801	2.232142857	6.90874036	17.36634036	23.74195989	22.02659816
Mean	42.36111111	91.28166915	118.116414	143.8175478	148.6560685	160.5446896
Gaussian	74.90335052	139.0035377	151.3317511	173.32527	176.7336633	189.3105203

Table.1 Results table

The speed up vs Image Size graph is shown in Fig.2

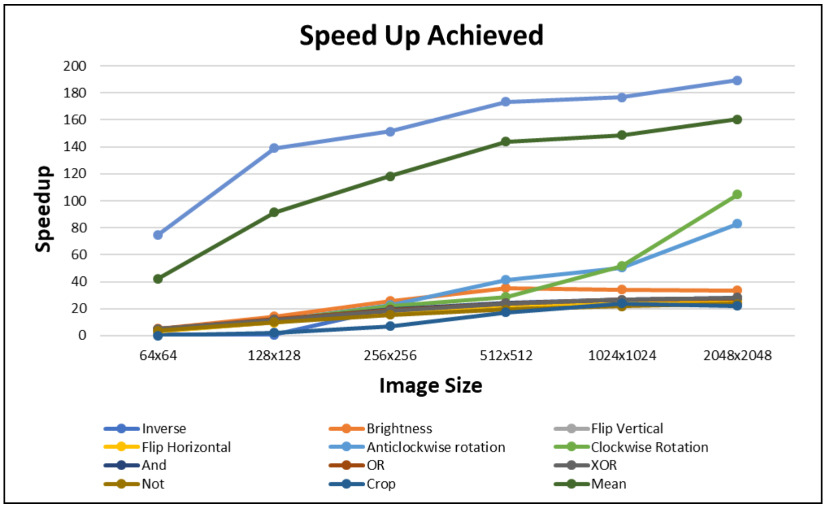

Fig.2 Speed-up graph

5 NEXT SET OF STEPS

We will try to optimize more on the already made operations. Moreover, we will be implementing some more operations till the final deadline. The more operations which will be implemented are: Contrast; LOCAL: Local(Edge Detection); Non-Linear (Median filter), Morphological (Erosion, Dilation); COLOR OPERATION (Channel split, Color picker, RGB to grayscale, Image to binary); HISTOGRAM EQUILIZERS; ADAPTIVE FILTER.

ACKNOWLEDGMENTS

The code for reading and writing the image was provided by Aradhya Neeraj Mathur, Teaching Assistant during the assignment.

REFERENCES

[1] https://books.google.co.in/books?hl=en&lr=&id=YumpCAAAQBAJ&oi=fnd&pg=PA1&dq=parallel+image+processing&ots=IbxY8iioXX&sig=wFjAYW35VxRq53G16mVhCYrlE0c#v=onepage&q&f=false.

[2] https://en.wikipedia.org/wiki/Volta_(microarchitecture).

[3] Gonzalez, Rafael C., Woods, Richard E., Digital Image Processing: Pearson 3rd edition.

[4] Soyata, Tolga., GPU Parallel Program Development Using CUDA: CRC Press, 2018 edition.