Parallelising Image transformations using OpenCL

—

By Utkarsh Gautam, Yogesh Kumar

- Many image transformations such edge detection, image sharpening etc. are essentially image convolution with different masks.
- Images can be partitioned into multiple patches and combinations of the convolutions of the patches is same as convolution of the entire image.
- We aim to use this property to speed up the convolution transformation for an image.

Experiment Details

Image Processing Algorithms

- rgb2gray
- Gaussian Blurring
- Gaussian Edge Detection

- Since each image transformation can be boiled down to convolution operations on image with a suitable mask
- For same operation, multiple masks may exist. Here we have restricted ourselves 3x3 sized masks
- We have used AMD Radeon 5000Series GPU on OpenCL to parallelize the algorithm

1/16	1/8	1/16
1/8	1/4	1/8
1/16	1/8	1/16

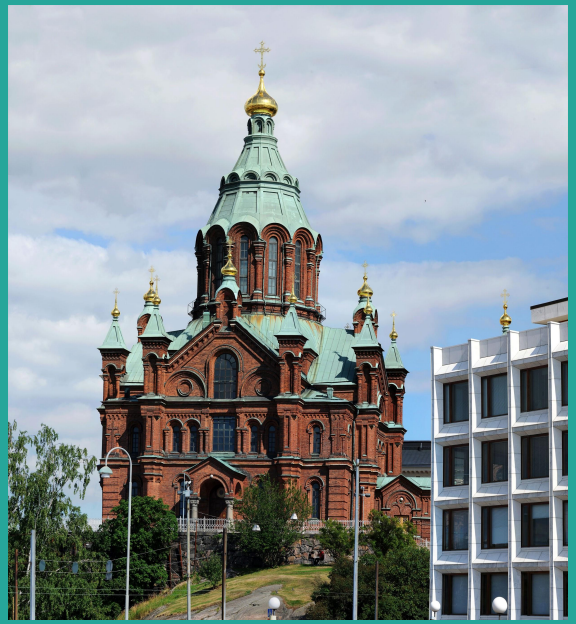
Mask for Gaussian Blurring

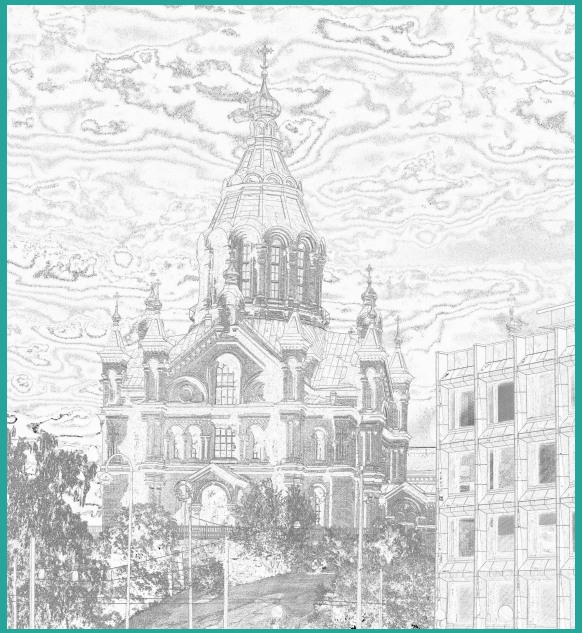
1	1	1
1	-8	1
1	1	1

-1	2	-1
2	-4	2
-1	2	-1

Mask for Gaussian Edge Detection

Results





Algorithm \ Image Size (in mb)	0.5	1	10	40
Serial rgb2gray	0.06238	0.1123	2.0568	3.4157
Serial gaussian blur	1.8490	3.8093	69.0476	111.4697
Serial gaussian edge detection	0.1947	0.39941	7.45562	11.9095
Parallel rgb2gray	0.0011	0.0014	0.01736	0.02435
Parallel gaussian blur	0.00099	0.00119	0.01237	0.01705
Parallel gaussian edge detection	0.00109	0.00142	0.01883	0.02835

Table: Time taken in seconds for each algorithm.
*Note this time is just (result matrix) computation time in the program, not the overall time.

Conclusion

We observe that with increasing image size (and increasing computational task) parallel algorithms perform way better than their serial parts.

Future Work

Convolution operation presented here is restricted to 2D filters. It can very easily be extended to 3D filters which can convolve over multiple channel images. This can be extended to faster neural network implementations in machine learning domain.

Thank you ! Questions ?