**Problem definition:**

This problem is automatic contrast enhancement on GPU. This is done by getting the minimum & maximum values of the image, subtracting minimum value from all images and then multiplying all pixels with 255.0/(maximum-minimum). This is done using thrust library.

**Algorithm Description:**

*(the algorithm checks for CUDA errors after each CUDA operation)*

*There are more explanations on functions themselves.*

* Copy the image to device
* Go through all pixels and find minimum and maximum values with parallel reduction
  + In my implementation I first put smaller values to first half of the image, bigger values to second half of the image. Then I find minimum and maximum in a joint kernel call (named kernel1 & CPUfunc1).
    - In kernel1 the first pixel of the image becomes the minimum, the middle pixel (in this case 512\*512/2’th pixel) becomes the maximum.
* Go through all pixels and subtract minimum from them (named kernel2 & CPUfunc2).
* Go through all pixels and multiply them with 255.0/(maximum – minimum) (named kernel3 & CPUfunc3).
  + In my implementation I don’t need to use integer result scaling since I set my function so that it can take more than 1 Npp8u integer.
* Output the image to a .pgm file.
* Free the pointers.

**Benchmarking:**

Results are included in benchmark.txt

deviceQueryResult.txt is included.

I failed to successfully implement GPU version of the code, only CPU version is included.

**Pros-cons:**

* Using CPP templates could help performance.
* Slower than non-thrust implementation (discussed in discussion).
* Smaller code.
* Easier CPU implementation.
* Harder(?) GPU implementation (discussed in discussion).

**Discussion:**

Thrust implementation was much much slower than my non-thrust implementation for CPU, and I couldn’t implement GPU version which suggests I can’t use thrust correctly or thrust is not good which is highly unlikely. I tried many different cases to make GPU version work but after hours of work I stopped (you can even see some remnants of tried code in all files). This is a clear sign that thrust is not as user-friendly as regular CUDA code when you first start using it. I must learn the correct ways of thrust for both CPU & GPU so please share the correct code(s) with me.

Also note that I didn’t really tried to make the code prettier since it doesn’t work correctly at all.

My most frequent errors were:

* thrust::detail::STATIC\_ASSERTION\_FAILURE<false>
* Compiler not being able to cast Npp8u & Npp8u\* correctly for thrust functions

I still don’t know the exact reason why CPU implementation works(slow but still works) and GPU implementation doesn’t work with similar fashion. Please explain this too.

**Environment:**

*DeviceQueryResult is included as a text file*

NVIDIA GeForce GTX 1060 3GB (compute capability 6.1)

Intel Core-I3 7100 @3.9GHz \w 8GB DDR4 RAM

Windows 10 Education Version 1809 (OS Build 17763.55)

Nvcc V10.0.130 compiler / Visual studio 2017 Community

**NOTE:** I didn’t had nppi.lib nor npp.lib, so I added all npp directories (cuda 10).