HPC - 2. Assignment - Parallel histogram Equalization using CUDA

Kristjan Kostanjšek, Nejc Ločičnik

I. Introduction

TODO

II. Optimizations and Results

The following subsections detail the optimizations applied to the sequential code, along with the computational speed-up achieved for each optimization. Each optimization is looked at in isolation. This means that the reported optimization speed-ups are for that specific part of the algorithm and not for the full exection. The full exectution speed-ups are reported at the end. The speed-up is calculated using the formula: $S = t_S/t_P$, where t_S represents the execution time of the sequential program, and t_P denotes the execution time of the parallel program. All reported times and speed-ups are averages of five runs.

A. RGB to YUV Conversion and Luminance Histogram Computation

TODO: - RGB to YUV is straightforward since pixels are independent - 1D thread scheme (no need for 2D since we aren't working with pixel surroundings, also more efficient thread use for irregular image sizes) - luminance histogram is calculated through shared memory - 2 different approaches with shared memory, 1. every block has a single shared array and we combine them into global memory at the end, 2. every warp (32 threads) has a shared array that then gets combined into a shared block array that then gets combined into global memory

TODO: Result table

TODO: Result comments

B. Cumulative Histogram Computation

TODO: - explain work-efficient optimization - temporary histogram is in shared memory, all 256 threads are in the same block

TODO: Result table

TODO: Result comments

C. New Luminance Computation

TODO: - every value is independent so again very straightforward optimization - 256 threads of the same block

TODO: Result table

TODO: Result comments

D. Assign New Luminance and YUV to RGB Conversion

TODO: - again very straightforward optimization approach since pixels are independent - 2 thread schemes comparison, 1D vs 2D, checking if 2D is any better since pixel locality might use the same values (easier to cache, but the histogram is an array of length 256 and should fit in the cache either way)

TODO: Result table

TODO: Result comments

E. Final results

TODO: - speed ups of full algorithm execution sequential vs. our best performing individual optimizations

TODO: Result table

TODO: Result comments

III. Conclusion

TODO