Project 4 Report

Zijian Feng, SID: 916024691
Yuxuan Huang, SID: 916509132

Part 0 Problems with using testing script

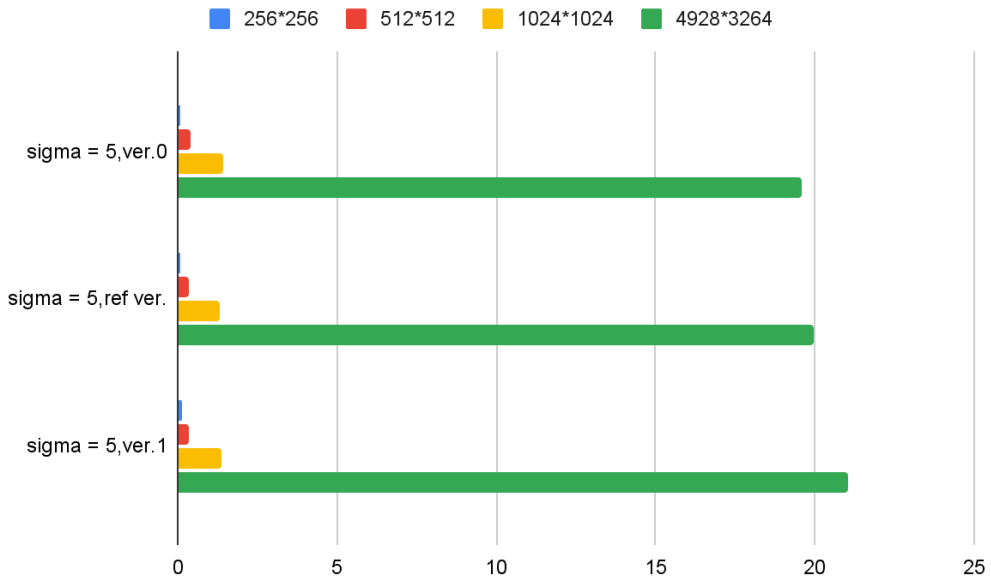
When we are testing our programs using the grading script. Though the required files and programs exist, the same errors will show for the eight correctness test cases--“Fail: got 'compare-im6.q16: unable to open image `name_size_sigma.pgm': No such file or directory @ error/blob.c/OpenBlob/2701.' but expected '0'”.

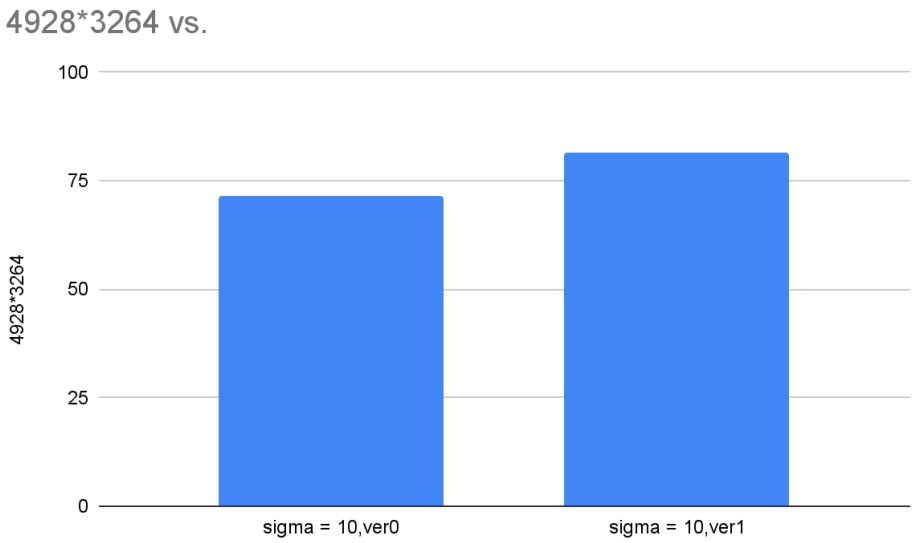
We first thought it was caused by us not naming the output files right but when we manually generated those files with the correct arguments, the test cases would be able to run and grade accordingly with no fail messages like above. Also, if we upload our source codes to gradescope, it can also grade without complaining, which also seems to run the grading script with command line arguments. Yet it seems that we are the only group that is encountering this issue so we are not sure if this is attributed to our code or the grading script.

Part 1 Serial Version

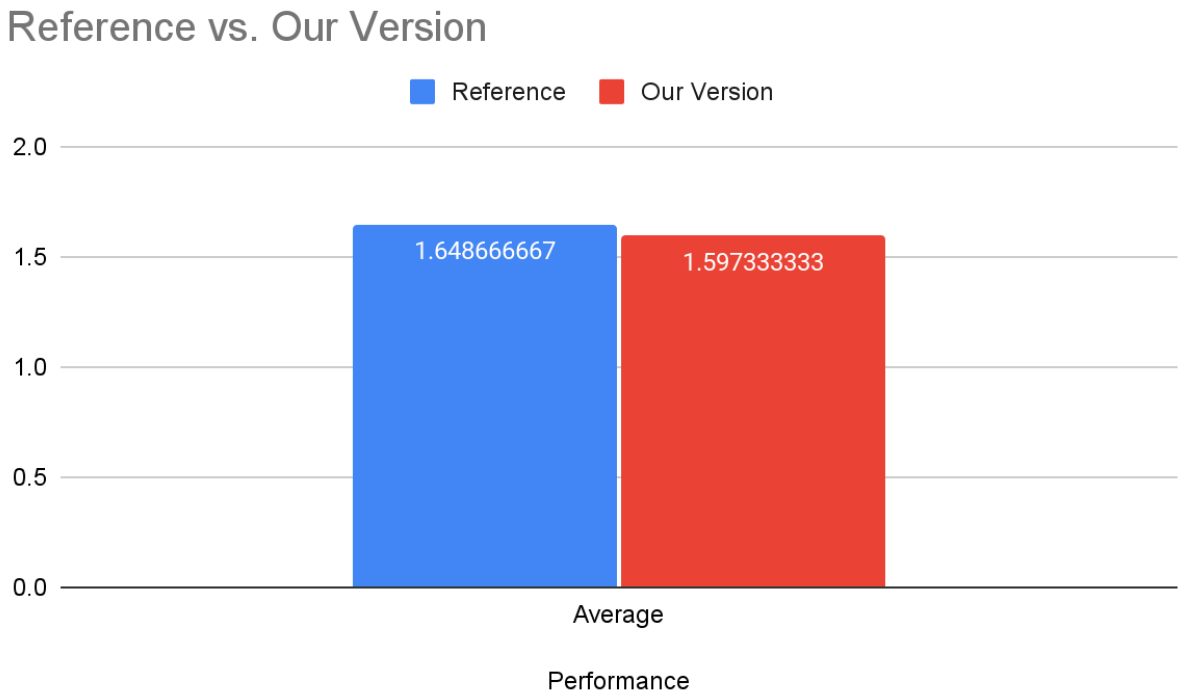
The basic operation goes like this: main gets the data in to memory, a loop will go through the pixels and for each pixel and another loop will do the convolution around that pixel. Finally the convoluted pixels will be written. In serial version, the first loop will be executed sequentially, yet the second loop, the inner loop, can be programmed into 2 ways: one is the old fashion 1 outer and 1 nested loop of which can allow variables, and the other is making only one loop but iterate until the multiple of the indices of both loop, and the data position will be calculated at each iterations. Though it was not thoroughly tested, the grader indicates that it is around 4 times slower than the 2-loop version. This is potentially caused by the lack of caching for the data position variables,x and y, for indexing each access to pixel data. Each time the indices will be recalculated that it gives caching heuristics little chance at guessing what data should be prefetched.

On the other hand, the 2-loop version also has 2 variances one though one is supposed to be less efficient than the other by calculating the variables inferred from the outer loops inside the inner loops. Ver0 is the more efficient one, Ver1 is the other.





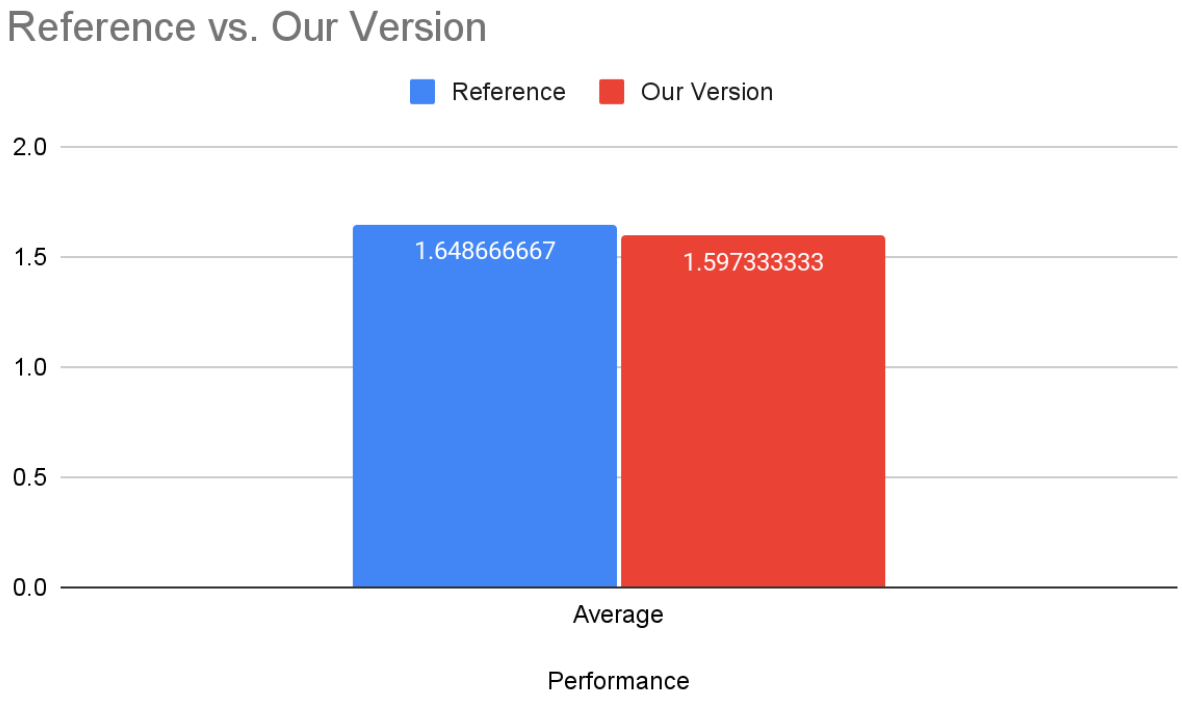
And we can see that when sigma is small there is also a similar growth rate for both 2-loop versions.But after all, ver0. calculates less so that the performance difference will be more noticeable when iteration counts are larger, shown in the vertical bar chart. With that in mind, choosing 2-loop ver0 gave us the best results.



Part 2 Cuda Version

Unfortunately, we were unable to get correct results for the cuda version. All images seem to be a bit darker compared to the right image. The only reason for this result is miscalculation on individual threads. But since we implemented the cuda version based on the serial version's logic, and the shortage of time, we were not able to find the problem and fix it. The only correct image we got was with ./gaussian_blur_serial ucd_pavilion_15mp.pgm ucd_pavilion_15mp_1.pgm 1.

We still did the performance test for the cuda version, all tests are performed with the script mention above.



Our version was able to match the reference's performance most of the time. To achieve this, we tried to assign optimal block size to improve efficiency. Besides that, we also practiced methods such as increasing locality to further improve the performance.