MPI

To parallelize the code with MPI, I assumed the program wound be run with 4 processes every time, so I gave each rank one filter size, so they could do the iterations in parallel. Then each rank sends its image to the rank below, rank 0 does not send one. Then the lowest 3 ranks finalize one image each, then write those to their respective files. To send an image between ranks, I make an MPI-datatype for the AccuratePixel struct, and send the data separately from the dimensions.

The timing of my MPI implementation gave these results:

Real: 16.185s
User: 16.966s
Sys: 0.104s

Open MP

TO parallelize the code for this problem, I used an OMP parallel region for practically the entire body of the Iteration function with only two shared variables computed outside this region. I used four threads, giving each thread one fourth of the height of the image to work with. Very little code had to be changed, Each thread that does not deal with the top line needs to initialize its own thread buffer in the middle of the image based on lines both above and below.

When ran on CMB, my Open MP implementation got those results:

Time(s): 1.37
Energy(j): 6.92
EDP(js): 9.48

CUDA

The parallelization using CUDA boiled down to splitting up the image into a three dimensional grid of blocks where each block has 32 threads. The dimensions of this grid are 60*40*30, so that the x dimension corresponds to a line of the image. The rows of the image are represented by the two remaining dimensions of the grid.

The timing of my CUDA implementation gave these results:

Real: 4.250s
User: 1.949s
Sys: 0.467s