ECE8780: Hackathon 2

David Langbehn (dlangbe@g.clemson.edu)
Instructor: Dr. Melissa C Smith

February 2021

---

1 Accelerating Convolution

You may have seen big softwares like photoshop perform blurring as the mouse moves over the image in *real time*. We've performed one optimization already in our code. We now look at ways to get the maximum performance out of our blurring kernel.

1.1 Part A: Shared Memory Kernel

This is a straightforward extension of the original algorithm (if you did not already do it in your original assignment). Extend the kernel you wrote in the lab to incorporate shared memory. Specifically, setup a shared memory of `TILE WIDTH` and use it to compute local sums. Compare and contrast the performance with the original kernel (lets call it a naive kernel) and the shared memory kernel. *Provide visual proof of performance improvement.*

1.2 Part B: Separable Convolution

Up until now we've been treating convolution as a weighted sum over a 2-D region. In practice (for example in Convolutional Neural Networks) this ends up as a *computationally intensive* task. Observe, that convolution is a separable operation i.e. you can independently convolve along the x and y dimension. Implement kernels `gaussian blur separable row` that computes the convolution along the row and a corresponding convolution that computes convolution along columns. You may use shared memory if that suits the need of the application. Compare the performance with the naive and shared kernel convolution.