CS 677/ HW #4 Report/ Theodore Jagodits

1.

So, in the naïve version, we would directly pull from global memory every time we need a variable. This leads to 6 global memory accesses per filter and then per thread. So, there are 12*xsize*ysize accesses per picture. In my version, I tile 16x16 thread blocks of shared memory and perform operations on the inner 14x14. That comes out to 16x16-14x14 more global accesses on the edges. But every inner tile uses all shared memory which makes it a lot faster. Each filter is a 3x3 so (14^2 * 3^2)/ (16+3-1) ^2 = about 5.4x reduction in bandwidth per filter.

2.

I selected a 2D structure for the blocks just so it is easier to understand the x and y dimensions of the block. I could have also used 1D and scaled it to 2D, it makes no difference.

3.

I already spoke about it in part 1, I use a 16x16 block to load all the tiles in for a 16x16 area, and then compute only the inner 14x14 matrix. The only issue is to make sure that the computed 14x14 outputs matched the input coordinates and they were right next to each other.