159.735 Sobel edge detection

Tim McMullen - 06222757

CPU vs GPGPU

Sequential.

The sequential program performed adequately, as it was able to run the sobel operations to create the correct edges, the time taken was a few seconds on the larger images. the results are as expected. The program runs by using a 9x9 matrix which goes through each pixel and changes it value based on the surrounding pixels and itself.

Cuda.

To impalement this program in Cuda first the image is opened and the sizes read, once that has been done memory is allocated on the host and then the graphics card using cudaMalloc, to store the image and the results. The image data is loaded into a buffer on the host then sent to the allocated memory on the graphics card, once there the same algorithm used in the sequential version is run on the image, but as it is now on the graphics card it is no longer needed to be looped through due to the streaming processors that are able to all work on the same data at once, dramatically increasing over all performance, and reducing the time required to find the edges. Once the sobel operation is completed the data is then sent back the the host to be written to a new file.

Results

BikesGray, (640x480)

Method	#Calls	GPU usec	%GPU time
Sobel	1	2628.67	91.27
MemcopyHtoD	1	34.304	1.19
MemcopyDtoH	1	316.865	7.53

Thalia (1600x1200

Sobel	1	16310	63.4
MemcopyHtoD	1	3014.18	11.71
MemcopyDtoH	1	6399.3	24.87

aubry_moon_zoom.fits (3150x4000)

Sobel	1	108888	63.26
MemcopyHtoD	1	19631.9	11.4
MemcopyDtoH	1	43606.9	25.33

Overall performance difference.

File	Thalia.fits (1600x1200)	Bikesgray.fits (640x480)	statesupt.fits (2214x3322)	aubry_moon.fits (3150x4000)
Cuda	0.334s	0.279s	0.49s	0.64s
Sequential	0.837s	0.143s	3.217s	5.546s

The Cuda and Sequential implementation of this program when compared show that on the smaller resolutions the sequential implementation is in fact faster than the cuda version, yet as the images become larger the benefits of using cuda are shown. the reason that the cuda implementation was slower on Bkiesgray is due the the communication over head, as it takes longer for the data to be sent and worked on at the GPU then received than for it to be worked on locally. yet as images become larger the benefit of using cuda is shown as it dramatically increases the performance and thus reduces time taken to find the edges of an image.

the difference between working on a single CPU vs working on a GPU is that while a CPU may have 4 processors, a GPU may have 400+ which all share memory allowing for each processor to work on the same data at once, so as shown by the results table once the communication over head has been surpassed in terms of significance the cuda implementation will continue to out perform a sequential version at an ever greater rate in regards to larger images