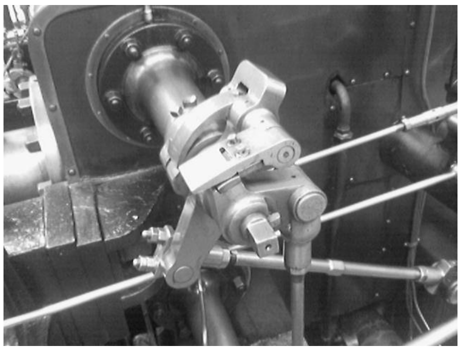
CUDA exercise 3: Sobel Filter

In this exercise the Sobel Filter – used for edge detection in image processing – should be implemented on the GPU.
`http://en.wikipedia.org/wiki/Sobel_operator`


Original image

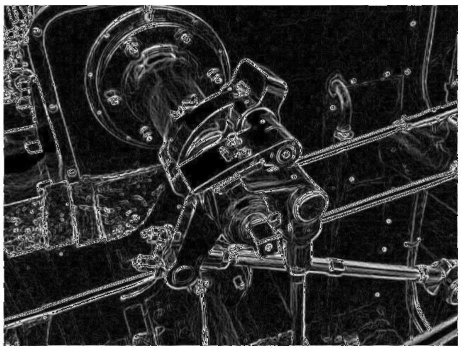


Image after sobel filter

1 Preliminary

Write the memory allocation on the device, the copy operations, and the allocation of the timers. Write the calls to the timers and the calculation of the speedups.

2 Kernel - Version 1

Implement the following parts of the kernel:

- loading of the pixels from the global memory
- Computing Gx and Gy.
- Writing the result back to global memory.

3 Kernel - Version 2

Implement the following parts of the kernel:

- Loading of the pixels to the shared memory.
- Setting the variables in which the neighboring pixels will are stored to the right values, loading either from the shared memory or from the global memory
- Computing Gx and Gy.
- Writing the result back to global memory.

Do not forget to synchronize after writing to the shared memory. (`__syncthreads();`)

4 Kernel - Version 3

Make a copy of the first kernel and implement the following parts of the second kernel:

- Change the size of the shared memory array.
- Loading of the pixels to the shared memory.
- Loading of the extra pixels on the border to the shared memory.
- Computing Gx and Gy.

Compare the 2 versions: speedups and number of registers used (look at the output of the compiler).