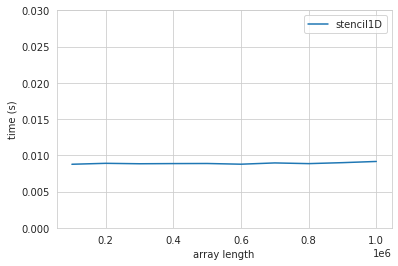
**ASIGNATURA Computación de altas prestaciones**

**Task 2**

**GPU Programming**

You must write a report answering the questions proposed in each exercise, plus the requested files. Submit a zip file through Moodle. Check submission date in Moodle (deadline is until 11:59 pm of that date).

* Exercise 1:
  + Compare the execution time for different values of N (array size): from 100.000 to 1.000.000 in steps of 100.000. Plot the result in a graph. Explain the results.



The time to execute is more or less the same for all the array lengths.

The GPU just utilizes more blocks/threads so we can do more computation in the same time.

If the workload is so high that the GPU is 100% utilized the length of the array would make a difference in the execution time.

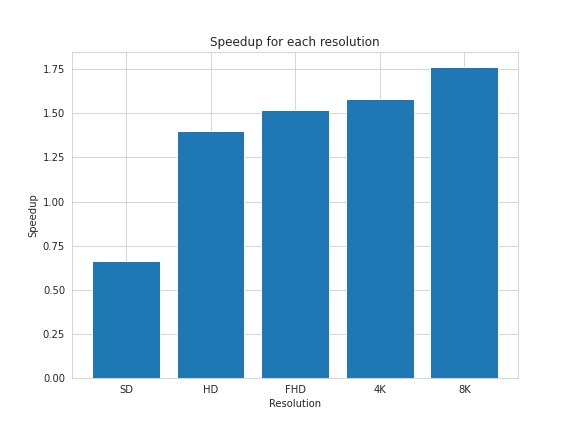
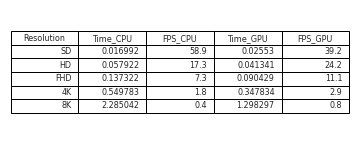
* + What BLOCK SIZE (number of threads per block) have you used? Do you think it is the most optimal? Explain.

We used a Block Size of 8.

Optimal would be a fraction of the arraylength.

I don’t think the actual size matters because modifying the BLOCK\_SIZE will also change the number of threads working on one block so that it will perform equally well.

* Exercise 2:
  + Fill in a table with time and speedup results compared to your manually vectorized CPU code for images of different resolutions (SD, HD, FHD, UHD-4k, UHD-8k). You must include a column with the fps at which the program would process. Discuss the results.



Comment:

When the Blocksize was bigger than 6, the SD image would not be converted properly.

Why?

With increasing amount of data, the advantage of the GPU is higher.

For small amounts of data, it is not feasible to copy everything to the GPU and run the kernel.

The GPU is superior in handling big amounts of data.

* + Explain how you implement the algorithm to be optimal for GPU

The Image array will be split into blocks by the GPU, our Blocksize is 16.

Then it will run multiple threads to compute the greyvalue for each pixel in the block

We use the possibility of Cuda to handle multidimensional arrays so that we can index the pixels by their corresponding x and y value.