David Pérez // NIA: 241375 Eduard Rodríguez // NIA: 242710

Universitat Pompeu Fabra Barcelona

Lab 3: Image Processing

This lab session consisted in reading an image from a text file and applying several different transformations to it using the GPU. This was done by communicating the host(CPU) and device(GPU) to interchange information to be used. The image was passed by the host which read the image in order for the device to process it. Hence, the GPU didn't have to resend the image, and it only had to send the variations of said image.

Exercise 1:

This part was the easiest. In this case, we simply had to establish a data region where the image was sent (without being resent) and the other images (which had been previously defined in CPU memory) were sent to be filled and resent with the final values.

In the functions to be executed by the GPU, we had to implement the `#pragma acc (...) ` directive with information about the use of the variables (where in all cases, the image was copied and the variation was completed and resent.

The whole process was around 0,03 seconds long.

Exercise 2:

In this part, our goal was to:

- Parallelize the code: done so by way of setting the different GPU implementations of the filters to a different async queue (invert → 1, smooth → 2 , …)

- No data management in the data region definition: which forces us to send the data in an unstructured manner. However, it still didn't differ much from the first one since we simply had to copy the data of the image and allocate space for the variations and update the latter.

At the end, before sending it back to the host we had to wait for all of them to be finished. The time was around 0,05, which leads us to believe we are not using the full capacity of the GPU

Exercise 3:

This part was the most complicated. We had to divide the calling of the functions into 4 different parts. This is because we can only load a quarter of the image and the variation. Hence, we have to go call by call creating and deleting memory space in order to avoid using more memory than given.

We have encountered many problems for exercises 2 and 3. Mainly, they do not generate images because of a segmentation fault within the clauses (we believe) that lead us to not know how our code performs.