SECOND PRACTICAL WORK

MASSIVE COMPUTATION

2021

DATA SCIENCE AND ENGINEERING

DESIGN OF AN IMAGE FILTER FUNCTION, SCALABLE AND OPTIMISED FOR IN-GPU EXECUTION

1)

You have to design and implement a filter function, in an independent module, which complains the follow characteristics:

```
def image_filter( image: GPU array floating point 32 bits,
                  filter_mask: GPU array 2D floating point 32 bits
                  filtered_image: GPU array 2D floating point 32 bits,
                  NumRowsImg: Int32 Image Numrows,
                  NumColsImg: Int32 Image Numcolumns,
                  NumRowsFilter: Int32 Image NumRows filter mask,
                  NumColsFilter: Int32 Image NumCols filter mask
                  )
```

- Receive a layer of the original image, a filter definition (filter mask), destination filtered image, numrowsimg, numcols image are the size of the image, and numrowsfilter and numcolsfilter are the size of the filter mask
- You must use a tiled memory algorithm to improve the access to the memory section to process.
- You have to design tile memory size, how to fill it, and synchronize the tasks.
- The filtered image must be stored in global memory filtered_image
- The filter shape could be variable: 5x5, 5x1, 1x5, 3x3, 3x1, 1x3.
    - You should take care in the image boundaries and borders. When you evaluate the first or last columns or rows, and if the filter exceeds the boundaries, the last available row or column must be repeated.
    - The filters will be provided by the tester. The sum of all terms in the filter will not exceed 1 and could have positive and negative values.
- The images shape will be variable. Could have 1 or 3 layers. The values will be positive integer values, between 0 and 255.
    - You have to filter layer by layer.
    - Convert the data type from unsigned int8 (uint8) to signed float 32, filter the layer and reconvert the filtered image to uint8.
- The output image must be a filtered image, with the same shape and values between 0 and 255 and stored in the shared memory space.

- Your GPU kernel must use the less possible conditional branches to fill the memory tile, evaluate boundaries and fill the result memory image.

2) You have to fill the provided a notebook, following the instructions, and filled the cells to load your module or modules, and invokes you functions, using the provided example images and filter masks.

The code must be well commented, indicating the different steps you are executing.

You can change the images and filters definitions, to test the stability of your code.

Store the images in the Google Drive, to access to they from Google Colaboratory.

Your code will be tested with different images and filters in order to get your grade.