# Image Zooming and Upscaling Using CUDA

GPU Programming course
A.Y.2022/2023


Politecnico di Torino

Marchetti Laura [s294767]
di Gruttola Giardino Nicola [s298846]
Durando Luca [s303395]

# Contents

# 1 Introduction

In this paper we implement an upscaling algorithm in CUDA: It is a technique used to produce an enlarged picture from a given digital image while correcting the visual artifacts originated in the zooming process.

Our zooming algorithm works on RGB images in PPM format: it provides as output a zoomed picture, cut out from the original one passed through the command line, while simultaneously correcting its aliased behavior by implementing a convolution with a specific filter.

The user needs to set, through the command line, the RGB picture that has to be zoomed, the coordinates of the center of the selection zone and the side length of the selection mask which has a square odd shape.

It is also left to the user to specify the filter to be applied in the convolution: it's both possible to set out a custom kernel or to create a Gaussian filter specifying GaussLength and GaussSigma.

# 2 Related works

## 2.1 The Pixel Replication Algorithm

In our implementation of the zooming algorithm we develop the gpu version of the already existing algorithm named "Pixel replication" also known as the "Nearest neighbor interpolation".
As its name suggests, it replicates the neighboring pixels in order to increase them in order to enlarge the image: it creates new pixels from the already given ones.
In this method each pixel is replicated n times row wise and column wise: therefore the size of the final image corresponds to (. . . )
This Algorithm has the advantage of being a very simple technique in implementing the zooming technique but, on the other hand, as the zooming factor increased the resulting image got more blurred.

## 2.2 Image Filtering Convolution Algorithm

In order to implement the convolution of the zoomed image, with a specified filter, we had to manipulate the already known version of the "Image filtering through convolution" Algorithm. In the original implementation of the algorithm ..

# 3 Proposed Method

In this section we will give to the lecturer a detailed description of the proposed algorithm: some of the implemented procedures will be explained through pseudo-code. The algorithm works in successive stages as described in the following:

- Image reading
- Image cut out
- Image enlargement and filling
- Image enhancement

## 3.1 Image Cut-Out

The aim of this step is to select the part of the original image that has to be trimmed: the dimension of the cut area is passed through command line and the script subsequently calculates the dimension of image in output.
The measure of the side of the final image is computed choosing the largest multiple smaller than the smallest dimension between width and height of the original image.
The function version for the cpu is implemented as "zero order zooming" whereas the gpu version is implemented as "get cut out": both of them carry out the same logic explained down below:

$$img_out[tid] = img[starting_byte + row_offset + column_offset] \tag{1}$$

## 3.2 Image Enlargement and Filling

The function version for the cpu is implemented as "zero order zooming" whereas the gpu version is implemented as "scale GPU": both of them carry out the same logic explained down below

$$img_out[coordinate_inizio_bordo+row_dim_largest_img+column_dim_largest_img+color_offset] = img[row_offset_smaller_img+ \tag{2}$$

## 3.3 Image Enhancement

# 4 Experimental Results

# 5 Conclusions

In this paper we proposed a technique for zooming and enhancing a digital picture in RGB colors by exploiting the gpu parallelism.