2D Gaussian Filter for Image Processing: A Study

Tarun Tyagi
M. Tech Student
Department of Electronics Engineering
Ideal Institute of Technology, Ghaziabad

Vishal Mishra
Assistant Professor
Department of Electronics Engineering
Ideal Institute of Technology, Ghaziabad

Abstract

This paper presents the study of 2D Gaussian filter and its vitality in image processing domain. The smoothing of images using 2D Gaussian filter brings out the best outcomes as compared to the conventional filters used to the date. Furthermore, when it comes to real time implementation of filter used for the image processing; it becomes a quite daunting task for the designers as it requires high computational resources. The purpose of this study is to present the FPGA resource usage for different sizes of Gaussian Kernel; to provide a comparison between fixed-point and floating point implementations.

Keywords: Gaussian Filter, FPGA Implementation, Fixed Point, Real Time Implementation

I. INTRODUCTION

The implementation of digital signal processing algorithms and availability of the digital systems have become more widespread since last two decades due to easy availability of digital systems. Earlier analog processors were used to perform the signal processing due to unavailability of digital processors. The digital signal processing became feasible to be performed in real time in the recent times due to hardware implementation of the algorithms developed in signal processing. It is all because of the requirement of higher level computations in the signal processing especially for the real time applications.

The demands for the high level computation systems combined with the performance of the VLSI architectures which led to the development of VLSI Digital signal processors such as TMS320(1982) and DSP56001(1987). The developers are left with enormous specific architectures of DSP with the ease in development of VLSI designs. The most common and usually employable DSP techniques are the FFT computing, FIR and IIR digital filters. These techniques require the three basic operations i.e. multiplication, storage and addition and these operations can easily be performed with the VLSI oriented architectures for Digital signal processing architectures.

The architectures developed through the VLSI implementation for DSP applications generally make use of parallel processing, multiprocessing, array processors, RISC i.e. Reduced instruction set and pipelining for the very high processing. The architectures developed for the Digital signal processing applications are tested and brought to the real time implementation through VLSI only. The most commonly and preferably used hardware for the implementation of DSP algorithms in VLSI is the FPGA. FPGA implementation of the digital signal processing algorithms makes it possible to develop a VLSI architecture for the high computation processing and multiprocessing at the real time.

II. COMPARISON BETWEEN CPU, GPU AND FPGA

In 2006 Nvidia Corporation announced a new general purpose parallel computing architecture based on the GPGPU paradigm (General-Purpose Computing on Graphics Processing Units): CUDA (Compute Unified Device Architecture) [1]. CUDA is an architecture classified as GPGPU, and it is a category of the SPMD (single process, multiple data; or single program, multiple data) parallel programming, the model is based on the execution of the same program by different processors, supplied with different input data, without the strict coordination requirement among them that the SIMD (single instruction, multiple data) model imposes. As a central point to the model are the so called kernels: C-style functions that are parallel executed through multiple threads and, when called from the application, dynamically allocate a hierarchy processing structure specified by the user. Interchangeably with the execution of the kernels, portions of sequential code are usually inserted in a CUDA program flow. For this reason, it constitutes a heterogeneous programming model.

The CUDA model was conceived to implement the so called transparent scalability effectively, i.e., the ability of the programming model to adapt itself in the available hardware in such a way that more processors can be scalable without altering the algorithm and, at the same time, reduce the development time of parallel or heterogeneous solutions. All aforementioned model abstractions are particularly suitable and easily adapted to the field of digital image processing, given that many applications in this area operate in independent pixel by pixel or pixel window approach. Many years before the advent of the CUDA architecture, Xilinx in 1985 made available to the market the first FPGA chip [2].

The FPGA is basically, a highly customizable integrated chip that has been used in a variety of science fields, such as: digital signal processing, voice recognition, bioinformatics, computer vision, digital image processing and other applications that require...
high performance: real time systems and high performance computing. The comparison between CUDA and FPGA has been documented in various works in different applications domains.

Asano et al [3] compared the use of CUDA and FPGAs in image processing applications, namely two-dimensional filters, stereo vision and k-means clustering.

Che et al [4] compared their use in three applications algorithms: Gaussian Elimination, Data Encryption Standard (DES), Needleman-Wunsch; Kestur et al [5] developed a comparison for BLAS (Basic Linear Algebra Subroutines); Park et al [6] analyzed the performance of integer and floating-point algorithms and Weber et al [7] compared the architectures using a Quantum Monte Carlo Application. In this work, CUDA and a FPGA dedicated architecture will be used and compared on the implementation of the convolution, an operation often used for image processing.

### III. IMAGE FILTERING PROCESS

A grayscale image is represented by a matrix of pixels with values ranging from 0 to 255. Sending a [512 x 512] image to the FPGA requires converting that image into a vector of 2612444 elements as shown in Figure 1, where DATA is the pixel value and ADDR is the memory address of each pixel, respectively. To code this in VHDL, the values of the pixels are represented with 8 bits, being the color black 0x00 and the color white OxFF.

![Fig. 1: Conversion of an Image Matrix to Image Vector](image)

The convolution of an image with a [3 x 3] kernel requires extracting each [3 x 3] sub-image from the original image, from left to right, and from top to bottom. After that, each of those elements is multiplied with the kernel image and the resulting products are finally added together (through a multiply-accumulate -MAC- operation), returning a single value for the output pixel. Figure 2 illustrates the MAC process.

![Fig. 2: Extraction of a [3 x 3] sub-image and MAC operations](image)
As seen in Figure 2, the Row Buffers (which are implemented in a Dual RAM FIFO memory) are used to store the rows of the sub-image belonging to the [3 x 3] window neighborhood. The pixels from those rows are then stored in shift registers. If the convolution involves a [N x N] kernel, then the process would require N - 1 row buffers for complete parallelism. This process is similar to simultaneous FIR filtering.

IV. FUTURE SCOPE

The Gaussian filter is a 2D convolution operator which is used to smooth images and remove noise. Further it can be implemented on FPGA and the area can be optimized as a matter of improvement in the same. Apart from the area, the work can be extended to delay optimization as well.

V. CONCLUSION

The study of 2D-Gaussian filter is presented here. The Gaussian filter is used to filter the images to eliminate the noise from the images. The real time implementation of the Gaussian filter is of great essence to prove its worth. The FPGA implementation has proved to be the best amongst CPU or GPU.

REFERENCES