University of Mosul
College of Electronic Engineering

FPGA Implementation of Video Deblurring

Abdulaziz Hussein Marie

M.Sc. Dissertation in
Computer and Information Engineering

Supervised by

Dr Emad Atiya Khalaf

2018 A.C. 1439 A.H.

University of Mosul

College of Electronic Engineering

FPGA Implementation of Video Deblurring

A Thesis Submitted by

Abdulaziz Hussein Marie

To

The Council of College of Electronic Engineering

University of Mosul

In Partial Fulfillment of the Requirements

For the Degree of Master of Sciences

In

Computer and Information Engineering

Supervised by

Dr. Emad Atiya Khalaf

2018 A.C. 1439 A.H.

Supervisor's Certification

I certify that the dissertation entitled (**FPGA Implementation of Video Deblurring**) was prepared by **Abdulaziz Hussein Marie** under my supervision at the Department of Computer and Information Engineering, University of Mosul, as a partial requirement for the Master of Science Degree in Computer and Information Engineering.

Signature:

Name: Dr. Emad Atiya Khalaf

Department of Computer and Information Engineering

Date: / /2018

Report of Linguistic Reviewer

I certify that the linguistic reviewer of this dissertation was carried out by me and it is accepted linguistically and in expression.

Signature:

Name:

Date: / /2018

Report of the Head of Department

I certify that this dissertation was carried out in the Department of Computer and Information Engineering. I nominate it to be forwarded to discussion.

Signature:

Name: Assistant Prof Dr. Abdul Baree R. Sulaiman

Date: / /2018

Report of the Head of Postgraduate Studies Committee

According to the recommendations presented by the supervisor of this dissertation and the linguistic reviewer, I nominate this dissertation to be forwarded to discussion.

Signature:

Name: Assistant Prof Dr. Abdul Baree R. Sulaiman

Date: / /2018

Committee Certification

We the examining committee, certify that we have read this dissertation entitled (**FPGA Implementation of Video Deblurring**) and have examined the postgraduate student (**Abdulaziz Hussein Marie**) in its contents and that in our opinion; it meets the standards of a dissertation for the degree of Master of Science in Computer and Information Engineering.

Signature:
Name:
Head of committee
Date: / /2018

Signature:
Name:
Member
Date: / /2018

Signature:
Name:
Member
Date: / /2018

Signature:
Name: Dr. Emad Atiya Khalaf
Member and Supervisor
Date: / /2018

The college council, in its …… meeting on / /2018, has decided to award the degree of Master of Science in Computer and Information Engineering to the candidate.

Signature:
Name:
Dean of the College
Date: / /2018

Signature:
Name:
Council registrar
Date: / /2018

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude and thanks to my supervisor, ***Dr Emad Atiya Khalaf*** for his continuous guidance, helpful suggestions and constant encouragement throughout this work.

Thanks are due to the Dean of the Electronics Engineering College for his valuable assistance. My appreciation is extended to the Head and all members of Computer and Information Engineering Department for their support and assistance.

Finally, I would like to extend our sincere appreciation to my family for encouragements, support and patience throughout the duration of my graduate study.

Abstract

Recently, video processing becomes much more significant topic due to the advances in the information technology. In this thesis, a new video deblurring algorithm implementation based on FPGA is presented. The proposed algorithm is based on the fractional order differentiation and modified Van Cittert algorithm. Genetic algorithm as an optimization tool has been used to tune the parameters of the proposed algorithm. The tuning cost function is a fuzzy set of entropy and peak signal to noise ratio.

Matlab 2017b as a development environment has been used to design and implement GUI application for the proposed algorithm to evaluate the results and compare with the conventional integer order filters (Laplacian, Robert, Prewitt, and Sobel). Qualitative and quantitative metrics have been used to evaluate the proposed algorithm. An online questionnaire survey based on Google forms used for visual qualifications of the restored images. More than 125 people respond to the questionnaire. 88.4% of the people stated that the proposed algorithm is better than the other ones. Normalized absolute error, structural content, normalized cross-correlation, peak signal to noise ratio, mean square error, entropy, average difference, and maximum difference are used as quantitative metrics.

Xilinx ISE design suite 14.5 has been used in conjunction with Matlab in hardware implementation. Two design methodologies have been used in the FPGA implementation on Xilinx Spartan-6 (Xc6slx16) SP601 evaluation kit. The first is based on Xilinx system generator, whereas the other using HDL verifier through FPGA in the loop and Modalism cosimulation. The aspects to reduce the hardware utilization of the FPGA are discussed. 44%, 45%, 15%, and 41% of hardware reduction in number of registers, LUTs, IOB, and DSP48A coprocessor respectively have been achieved. Under real time considerations, a 128x128 frame size has been implemented in the first technique, whereas full high definition (1080x1920) in the other technique.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Name
AD	Average Difference
ASIC	Application Specific Integrated Circuits
CFE	Continuous Fraction Expansion
CLB	Configurable Logic Block
CPLD	Complex Programmable Logic Devices
DCT	Discrete Cosine Transform
DSP	Digital Signal Processor
EDA	Electronic Design Automation
FD	Fractional Differential
FIFO	First In First Out
FIL	FPGA in the Loop
FOD	Fractional Order Differentiation
FPGA	Field Programmable Gate Arrays
GA	Genetic Algorithm
G-L	Grunwald–Letnikov
GPU	Graphics Processing Units
GTS	Generalized soft Thresholding
HDL	Hardware Description Languages
ICs	Integrated Circuits
IP	Intellectual Property
ISE	Xilinx Integrated Software Environment
LUT	Lookup Tables
MD	Maximum Difference

MSE	Mean Square Error
MTF	Modulation Transfer Function
NAE	Normalized Absolute Error
NCC	Normalized Cross-Correlation
PSF	Point Spread Function
PSNR	Peak Signal to Noise Ratio
PWNLK	Pixel Wise Nonlinear Kernel
RAM	Random Access Memory
R-L	Riemann–Liouville
SC	Structural Content
SoC	System on Chip
TV	Total Variation
VLSI	Very High Scale Integration
XSG	Xilinx System Generator

Chapter One

Introduction and Literature Review

1.1 Introduction

In the recent years, video and image processing are becoming more important in the fields of research. Video is a stream of frames. Frame is an image which is a two dimensional array of picture elements, pixels or pels [1]. Image quality degradation may not be avoided during the video acquisition which caused by atmospheric turbulence, camera lens defocus and relative movement between the camera and the scene. The inverse issue of the blurring is called deblurring [2]. The main aim of deblurring is recovering an image which suffered from degradation [3]. Image deblurring techniques are encountered in a wide range of applications, such as microscopy, medical imaging, astronomy, remote sensing, super resolution, optics, motion tracking, and photography [4, 5, 6].

Implementation of such applications based on general-purpose computer may be more easier, but not efficiently in time because of the constraints on peripheral devices and memory. The implementation of application specific hardware gives a better speed than the implementation of software [7]. The advances in the very high scale integration (VLSI) technology made the hardware implementation becomes an attractive alternative. The implementation of complex computation on hardware by utilization of pipelining and parallel algorithms make a significant reduction in the time of execution. There are two kinds of technologies available for hardware

implementation. Fully custom hardware devices which known as application specific integrated circuits (ASIC) and semi-custom, which are programmable devices like field programmable gate arrays (FPGA). Fully custom ASIC offers highest performance design, but the cost and complexity of the design is very much. FPGAs are reconfigurable devices [8]. Hardware design techniques like pipelining and parallelism can be implemented on FPGA using electronic design automation (EDA) tools that can help the designer with design entry, hardware generation, test sequence generation, verification and design management [9, 10, 11]. FPGAs generally consist of a system of logic blocks and some amount of Memory, all wired together using a vast array of interconnections. All of the logic in an FPGA can be rewired, or reconfigured [12].

1.2 Review of Literature

Researchers have applied many methods and algorithms for image and video deblurring. Different approaches have been used in the hardware implementation of the deblurring techniques.

Jun Kong et al. in 2017 [13] proposed a blind deblurring algorithm using hyper-Laplacian prior for regularization of image gradients. Generalized soft thresholding (GTS) has been used to solve the problem of non-convex during the deblurring process. Qualitative and quantitative metrics have been used to qualify the proposed algorithm with popular approaches.

Hongyan Wang et al. in 2017 [14] presented a new bind deblurring method based on image prior using the regularization of elastic-net of singular values. Complex filtering is not required to select the salient edges. Results

showed that the proposed algorithm performance favorably against the popular algorithms.

Dong-Bok Lee et al. in 2013 [15] developed an iterative blind algorithm for video deblurring exploiting the unblurred neighborhood frames to estimate the blur kernel. The residual deconvolution has been presented to reduce the ringing of artifacts produced in conventional deconvolution.

Mohammed Alareqi et al. in 2017 [16] developed an FPGA implementation based Hardware co-simulation of image enhancement. The contrast stretching and thresholding have been used to enhance the image for biomedical applications. Xilinx system generator and Matlab Simulink have been used for co-simulation on Virtex-5 FPGA development kit.

A M Deshpande and S Patnaik in 2011 [17] presented a comparison study and performance evaluation of Wiener filtering, Richardson-Lucy algorithm, constrained least squares filtering, direct inverse filtering, and pseudo-inverse filtering. The objective evaluation involved peak signal to noise ratio (PSNR) and mean square error (MSE).

Yie-Fie Pu et al. in 2010 [18] implemented a fractional order differentiator masks for image enhancement. Six fractional order differential masks have been proposed. Experimental analysis proved that the fractional differentiator is better than conventional integer order differentiators.

Wenqi Ren et al. in 2017 [19] proposed a pixel wise nonlinear kernel (PWNLK) for video deblurring. Experimental results on blurred videos showed that PWNLK is better than state-of-art techniques.

M. Lopez-Ramirez et al. in 2017 [20] developed a novel and optimized method to extend the depth of filed in one image. The proposed approach has been implemented on Xilinx Virtex-6 and Altera Stratix-III. Experiments showed that FPGA implementation is best in performance than graphics processing units (GPU) and Digital Signal Processors (DSP).

Priyanka Raina et al. in 2017 [21] proposed a kernel estimation using hardware acceleration for application of image deblurring. 40nm CMOS technology has been used as a platform for fabrication of proposed model. The fabricated chip has been mounted on Xilinx Vertix-6 development kit. The developed model has 78X speed faster than intel Core i5 CPU for kernel estimation and 56X for the overall deblurring algorithm of Full HD images.

G. B. Reddy and K. Anusudha in 2016 [22] implemented image edge detection based Laplacian of Gaussian, Sobel, Prewitt, and Robert operators on Xilinx Spartan-3E FPGA development kit. Xilinx system generator and Matlab Simulink have been used as a platform for hardware implementation.

Wei Wang and Peizhong Lu in 2012 [2] developed a new deblurring algorithm by combining fractional order differentiation with Total Variation (TV). Results showed that the performance of the proposed method is better in visual quality and PSNR than conventional deblurring algorithms.

Chien-Cheng Tseng and Shyi-Chyi Cheng in 2012 [23] presented a color image sharpening algorithm based on discrete cosine transform (DCT) and fractional order differentiation.

Chien-Cheng Tseng and Su-Ling Lee in 2013 [24] presented a color image sharpening using fractional order differentiation and modulation transfer

function (MTF). YCbCr color space has been used in the implementation of the algorithm.

Yie-Fie Pu et al. in 2008 [25] presented an analytical study of fractional order differentiation in view of kinetics and information theory. The hardware model of fractional order differentiator mask has been developed for image processing applications. Experimental results proved that the fractional order differentiator has excellent capabilities for texture enhancement.

Thusitha Chandrapala et al. in 2012 [26] developed an FPGA implementation of vide deblurring. The hardware implementation consists of two soft cores in the FPGA chip, one for blur detection and the other for restoration. The deblurring system achieved 15 frame rate at 1280x720 HD video stream.

Zohair Al-Ameen et al. in 2012 [27] presented an extensive study of image deblurring algorithms. The algorithms are Van Cittert, enhanced Van Cittert, Poisson Map, Landweber, Richardson – Lucy, optimized Richardson – Lucy, and Laplacian sharpening filters.

Emrah ONAT in 2016 [28] presented a real time hardware implementation of Robert, Sobel, Laplacian, and Prewitt filters for edge detection. Xilinx Zynq 7000 has been used in the hardware implementation.

Qi Yang et al. in 2016 [29] presented fractional calculus definitions and discretization schemes. They introduced a survey of fractional calculus and it's applications in image enhancement, edge detection, denoising, segmentation, recognition, registration, fusion, compression, encryption, and restoration.

T. Singh and B. M. Singh in 2016 [30] presented image deblurring as a comparative analysis survey for more than twenty five researches. The comparative study based on PSNR as an objective assessment.

1.3 Aims of the Study

The following related points provide the main objectives of the current study:

1. To study the theory of video blurring and debluring.
2. To study the theory of fractional order differentiation and its applications in image processing.
3. To use genetic algorithm to search for the optimal values of the gain and order of the proposed two-dimensional fractional order differentiator (2D-FOD) debluring algorithm.
4. To develop the required software to implement and examine the proposed algorithm.
5. To implement a real time video debluring system based FPGA.

1.4 Thesis Layout

The mathematical model of video blurring, causes of video blurring and the debluring techniques are given in chapter two. Chapter three presents the theory of fractional order differentiation and the most famous two dimensional fractional order differentiators used in image processing. The software and problems associated with developing the required software for the proposed 2D-FOD debluring system are given in chapter four. Chapter five presents the FPGA implementation of the proposed debluring system. Performance analysis

of the implemented system is discussed in this chapter. Finally, conclusions and some suggestions for future researches are propose in chapter six.

Chapter Two

Image Deblurring

2.1 Introduction

Blur is a kind of bandwidth reduction during the image acquisition process, which may produce from deferent sources and usually makes bad quality vision. Occasionally blur may be arisen by the photographer to enhance picture's expressiveness, but unconsciously blur will degrade the picture quality. The mathematical model of blurring process is usually required to restore a higher image quality from the blurry version [31].

2.2 Mathematical Model of Image Blur

The mathematical model of the blurry image is computed by convoluting the original image with a blur kernel plus a noise. The mathematical model is given in equation (2.1). The blur kernel is also known as Point Spread Function (PSF). The PSF makes a pixel brightness to be spread over the pixels of the neighborhood [32].

$$g = f \otimes h + n \qquad (2.1)$$

Where g denotes the blurry image, f is the original image, h is the PSF, n is the noise, and the $\otimes$ symbol is the convolution operator. Equation (2.1) can be expressed in frequency domain as illustrated in equation (2.2) [33].

$$G = F \cdot H + N \qquad (2.2)$$

Where, $G, F, H, and\ N$ are the Fourier transform of $g, f, h, and\ n$ respectively. H is a low pass filter because it tries to discards the high

frequency information in the F [31]. Figure 2.1 visualizes the process of blurring.

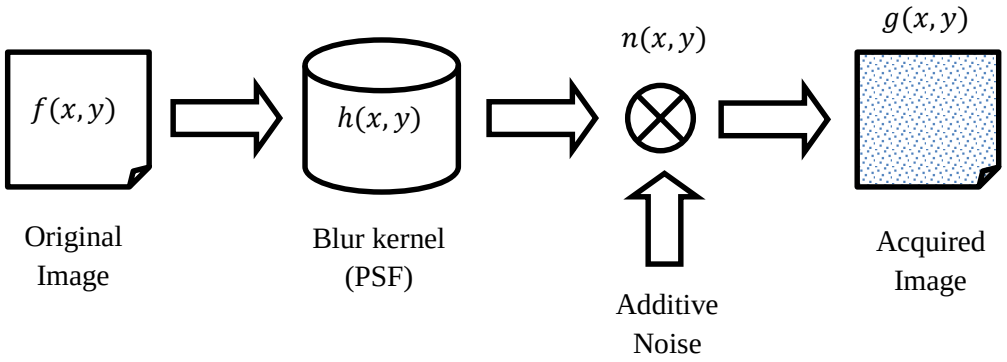


Fig 2.1: Image Blurring Process.

2.3 Causes of Image Blur

There are three main types of blurs respect to their physical properties: atmospheric blur, out of focus blur, and motion blur.

2.3.1 Atmospheric blur

Atmospheric blur comes from both optical turbulence and small angle scatter of light by aerosols. Optical turbulence is characterized by random refractions of light caused by spatiotemporal changes in atmospheric properties such as temperature and density.

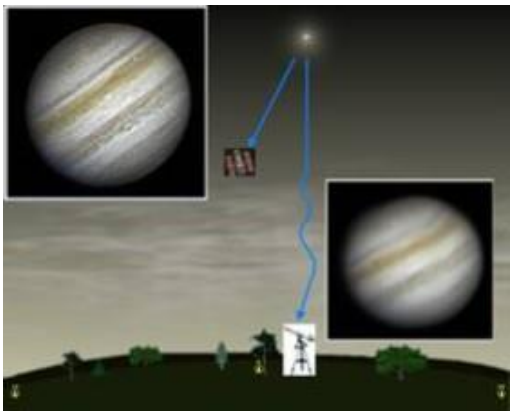


Fig 2.2: Atmospheric blur.

Chemical substances of variant sizes can also produce deferent kinds of scattering to light waves with variant waves length. This phenomenon is known as aerosol scattering in atmospheric sciences. In general, turbulence blur is more of a problem closer to the Earth's surface and aerosols blur is more prevalent at higher elevations. Both aerosol scattering and optical turbulence will introduce blur. Atmospheric blur is a considerable issue in satellite imaging, atmospheric science, and remote sensing [31].

2.3.2 Out of focus blur

The different depths of objects would cause image blur in digital cameras with automatic focusing system [34]. For instance, when capturing many objects in the scene with deferent ranges, the digital camera lens can focus on a single object or a region of interest whereas leaving the rest objects out of focus. Figure 2.3 illustrates that cameras have a finite depth of focus and the whole image can be perfectly in focus only if the whole scene is in the same distance from camera. A more widely used parametric model for out-of-focus blur is a circularly symmetric 2D Gaussian function [31].

$$K(x,y) = \frac{1}{2\pi\sigma^2} exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \qquad (2.3)$$

Fig 2.3: Out of focuse blur

2.3.3 Motion blur

Motion blur caused by a relative motion between the camera and a scene which is inevitable due to the nature of a camera sensor that accumulates incoming light during exposure time period [35,36]. Blurred images can be obtained by camera shaking or object motion during image acquisition processes [37]. Usually, the motion blur, which can be caused by the camera vibration when shutter is open, is known as camera motion blur, or camera shaking blur [30]. Camera shaking is a very prominent problem for photography, especially in low-light conditions [38].

Figure 2.4 shows examples of images that are blurred by three different motion paths.

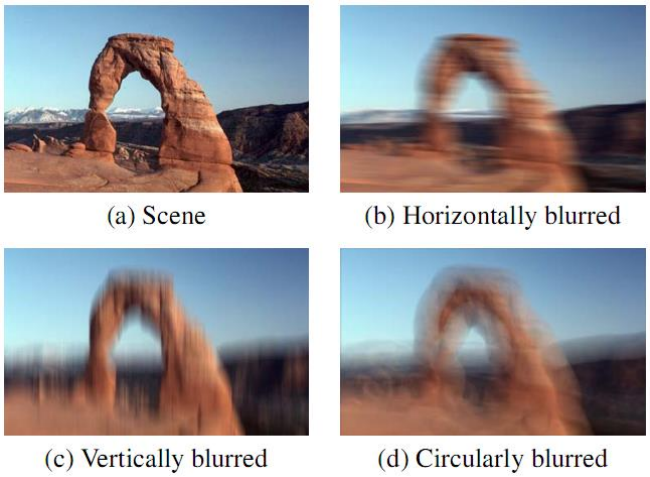


Fig 2.4: Different motion blur.

When camera is fixed over the period of exposure, any object moving with respect to a static background can also lead to another motion blur, which is known as object motion blur. Obviously, objects in the scene may not undergo the same motion relative to the camera. Figure 2.6 show two real examples of object motion blur [39, 40]. The blur is usually space-variant, nonlinear, local and multiple. Practically, the motion blur can be caused by both camera shaking and moving object [31].

Fig 2.5: Object motion blur.

2.4 Ill-posed Problem in Image Deblurring

Image deblurring is a well-known ill-posed inverse problem [41]. From equation (2.1), there are three unknown variables, The PSF, original image, and noise. The PSF must be estimated before we can compute the original image. Even if it is assumed that the PSF is known, the restoration process is still complex because it is an ill-posed problem, and any noise in the degraded images can be amplified dramatically in the restored images. Moreover, the denoising must be happened before the deblurring process to cancel the noise [32].

2.5 Deblurring Techniques

The deblurring techniques can be divided into blind and non-blind:

1. In non-blind deblurring algorithms, the PSF must be known [42]. Many of the non-blind deblurring algorithms use the inverse of the PSF in frequency domain such as wavelet transform [44], Wiener filter [43], and so on. Whereas the other non-blind deblurring algorithms works directly on spatial domain as in Bayesian inferences [45].

2. The blind deconvolution [46, 47] is the process of restoring a fine detail image from degraded version, when the blur kernel is unknown.

The Blind Deconvolution Algorithm can be used effectively when no information about the distortion (blurring and noise) is known. The algorithm restores the image and the point-spread function (PSF) simultaneously [47]. There are two typical approaches for blind deblurring problem. In the first approach, the blur identification procedure is realized in a separate step to estimate the blurring function. Then, any available deblurring method is used to estimate the original image. In the second approach, the blur identification and the image restoration procedure are incorporated in a unifying algorithm [48].

In this work a blind deconvolution has been used by applying a special domain filter such as first order differential gradient operator, second order differentiation (Laplacian) operator, and fractional Differentiation operator to restore sharpen image from degraded image.

2.6 Differential Gradient and Operators

In this section, we review integer-order differential gradient and operators, while the fractional order differential operator will be discussed in chapter three.

2.6.1The First Order Differential Operator

In image processing, first order differential is achieved through the gradient method. For the Function $f(x,y)$, the gradient of its coordinates (x,y) is through a two-dimensional column vector defined by:

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \qquad (2.4)$$

The vector model is given by:

$$\nabla f = \text{mag}(\nabla \text{f}) = [G_x^2 + G_y^2]^{\frac{1}{2}} = \left[\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2\right]^{\frac{1}{2}} \qquad (2.5)$$

There: $\frac{\partial f}{\partial x} = f(x+1, y) - f(x, y$, and $\frac{\partial f}{\partial y} = f(x, y+1) - f(x, y)$

For ease of calculation, the general method is convolution calculation. Firstly, we obtain a convolution mask, then using the mask to loop calculation on whole image. First-order operators, which include Roberts operator, Prewitt operator, Sobel operator, Robinson operator and Kirsch operators. This article chooses Prewitt and Sobel operator, as the first order differential operators of comparative experiments. Here are Prewitt and Sobel edge detection operator of the convolution mask [50->49].

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

(b) X-axis direction mask (a) Y-axis direction mask

Fig 2.6: Sobel mask operator.

The weight value 2 of Sobel operator is used to increase the importance of the center.

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

(a) X-axis direction mask (b) Y-axis direction mask

Fig 2.7: Prewitt mask operator.

2.6.2 The Second Order Differential Operators

Laplacian filter well-known filter used to sharpening images. Image deblurring is sharpening task. There are three types of Laplacian filter, -8, -4, and 9 operators. Figure 2.8 shows the three types of Laplacian operators.

0	1	0
1	-4	0
0	1	0

1	1	1
1	-8	1
1	1	1

-1	-1	-1
-1	9	-1
-1	-1	-1

Fig 2.8: Different Laplacian operators.

The sharpening formula using Laplacian (-8) and (-4) operators is given in equation 2.6.

$$F = g - [g \otimes L] \qquad (2.6)$$

Where, F is the sharpened image, g is the blurred image, L is Laplacian operator, and $\otimes$ is the convolution operator. Whereas, the sharpening formula using Laplacian (9) operator is given in equation 2.7.

$$F = g \otimes L \qquad (2.7)$$

The degree of the sharpening depends on the type of the Laplacian operator. The Laplacian operators (9) and (-8) sharpening the images better than the (-4) [27].

Chapter Three

Fractional Order Differentiation

3.1 Introduction

The conventional image enhancement algorithms are based on integer order differential mask operators. Recently, fractional calculus has been presented for various applications of science and engineering. The current chapter introduces the definition of fractional calculus and the discretization schemes in one and two-dimensional space for fractional order differentiation.

3.2 Start of Fractional Calculus

Fractional calculus is 300 years old but not well known among engineering and science communities. In 30 September 1695, L' Hopital wrote a letter to Leibniz asking him about his publication for *nth* order of fraction derivative $\frac{d^n g(x)}{d\,x^n}$, if $n = 1/2$. Leibniz replied "an apparent paradox from which one day useful consequences will be drawn". From these words fractional calculus was started [50, 51].

3.3 Definitions of Fractional Calculus

Fractional calculus is a generalization of differentiation and integration. The non-integer order operator ${}_aD_t^{\alpha}$, where $\alpha \in R$ in the interval$[a\,,t]$. The integro-differential operator can be defined as in equation 3.1 [52, 53].

$$ {}_aD_t^{\alpha} = \begin{cases} \dfrac{d^{\alpha}}{dt^{a}}\ , & \alpha > 0, \\ 1, & \alpha = 0, \\ \displaystyle\int_a^t d(t)^{\alpha}\,, & \alpha < 0. \end{cases} \qquad (3.1) $$

The following are the popular definitions of fractional order differentiation [50].

1. Riemann–Liouville (R-L)

$$ {}_aD_t^{\alpha} g(t) = \frac{1}{\Gamma(n-\alpha)} \left(\frac{d}{dt}\right)^n \int_a^t \frac{g(\tau)}{(t-\tau)^{\alpha-n+1}}\, d\tau\ \ , (n-1) < \alpha < n \quad (3.2) $$

Where n is integer and α is real number.

The fractional order derivatives of some standard functions using Riemann–Liouvelle (R-L) are presented in Table 3.1 [50]. Figure 3.1 shows the fractional derivative of sine wave at different orders.

2. Grunwald–Letnikov (G-L)

$$ {}_aD_t^{\alpha} g(t) = \lim_{h \to 0} \frac{1}{h^{\alpha}} \sum_{j=0}^{\left[\frac{t-a}{h}\right]} (-1)^j \binom{\alpha}{j} g(t-jh) \qquad (3.3) $$

Where $\left[\frac{t-a}{h}\right]$ $is\ an\ integar\ number.$

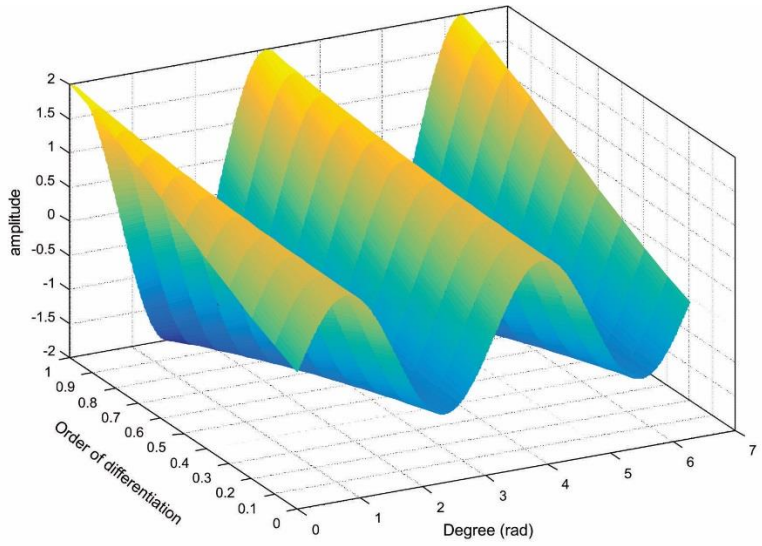
3. M. Caputo

$$ {}_a^C D_t^{\alpha}\ g(t) = \frac{1}{\Gamma(n-\alpha)} \int_a^t \frac{g^{(n)}(\tau)}{(t-\tau)^{\alpha+1-n}}\, d\tau, \quad (n-1) \le a < n, \quad (3.4) $$

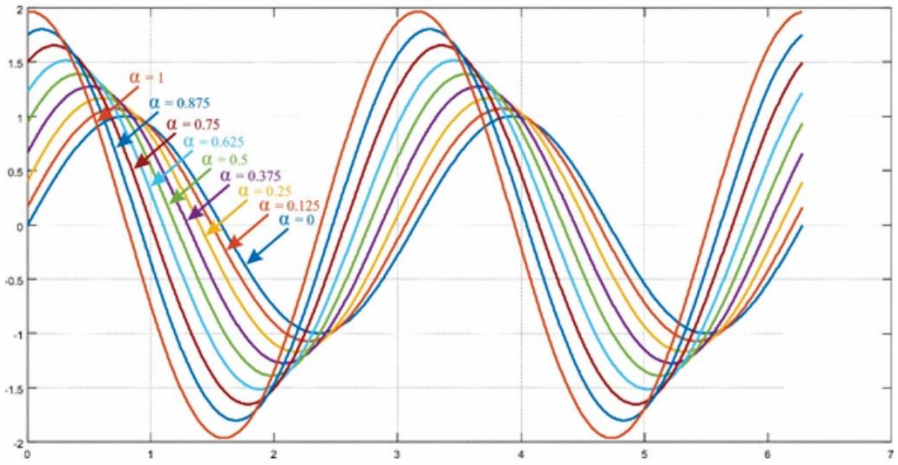
Where n is integer and α is real number.

Table 3.1: Fractional derivative for some standard functions [50].

Function $g(t)$	${}_0D_t^{\alpha} g(t)$. Fractional Derivative
$e^{\lambda t}$	$\lambda^{\alpha} e^{\lambda t}$
$\sin \lambda t$	$\lambda^{\alpha} \sin\left(\lambda t + \frac{\pi\alpha}{2}\right)$
$\cos \lambda t$	$\lambda^{\alpha} \cos\left(\lambda t + \frac{\pi\alpha}{2}\right)$



(a) Three dimensions view.



(b) Two dimensions view

Fig 3.1: Fractional derivative of sine wave at different orders.

3.4 Fractional Order Differentiation Discretization

The main approach for presenting image processing applications based fractional order differentiation is demonstrated in figure (3.2) [29, 54, 55]. The main methods of discretization are given bellow:

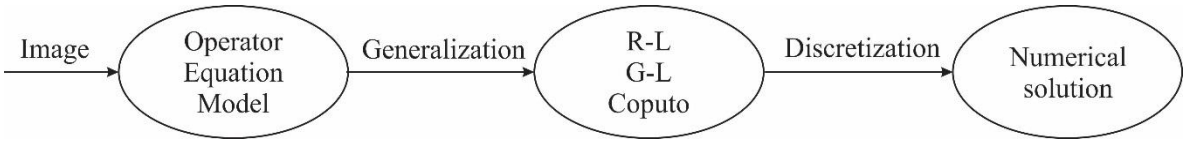


Fig 3.2: Image processing based FOD [29]**.**

1. Tustin Transformation:

The discretization is based on Tustin discretization [56, 57].

$$D^{\pm\alpha} = \left(\frac{2}{T} \cdot \frac{(1 - Z^{-1})}{(1 + Z^{-1})}\right)^{\pm\alpha} \qquad (3.5)$$

For a given (n) order of approximation:

$$D^{\alpha} = \frac{2}{T} \lim_{n\to\infty} \left(\frac{A_n(Z^{-1}, \alpha)}{A_n(Z^{-1}, -\alpha)}\right) \qquad (3.6)$$

Where:

$A_0(Z^{-1}, \alpha) = 1$

$A_n(Z^{-1}, \alpha) = A_{n-1}(Z^{-1}, \alpha) - C_n Z^{-n} A_{n-1}(Z^{-1}, \alpha)$

$$C_n = \begin{cases} \alpha/n & n\ is\ odd \\ 0 & n\ is\ even \end{cases}$$

2. Discretization using AL-ALaoui operator.

$$D^{\pm\alpha} = \left(\frac{8}{7T} \cdot \frac{(1 - Z^{-1})}{(1 + Z^{-1}/7)}\right)^{\pm\alpha} \qquad (3.7)$$

This equation can be approximated using continuous fraction expansion (CFE) [58].

3. G-L discretization [51].

$$g^1(x) = \lim_{h\to 0} \frac{g(x+h) - g(x)}{h}$$

$$g^2(x) = \lim_{h\to 0} \frac{g^1(x+h) - g^1(x)}{h}$$

$$g^2(x) = \lim_{h\to 0} \frac{g(x+2h) - 2g(x+h) + g(x)}{h^2}$$

$$g^n(x) = \lim_{h\to 0} \frac{1}{h^n} \cdot \sum_{m=0}^{n} (-1)^m \binom{n}{m} g(x-mh)$$

Where

$$\binom{n}{m} = \frac{n!}{m!\,(n-m)!}$$

For non integer n, i.e., α. $\binom{n}{m}$ can be replaced using Gamma function $\frac{\Gamma(\alpha+1)}{m!\Gamma(\alpha-m+1)}$

$$g^\alpha(x) = \lim_{h\to 0} \frac{1}{h^\alpha} \cdot \sum_{m=0}^{\left[\frac{x-\alpha}{h}\right]} (-1)^m \frac{\Gamma(\alpha+1)}{m!\,\Gamma(\alpha-m+1)} g(x-mh) \quad (3.8)$$

3.5 Two Dimensional FOD Masks

In digital image processing the pixels are processed in eight symmetrical directions to enhance pixels antirotation ability. The directions are positive-X, left up diagonal, positive-Y, right up diagonal, negative-X, right down diagonal, negative-Y, and left down diagonal as shown in figure (3.3) [25].

Yi-Fie proposed six masks to be used in image processing. He mentioned that the YiFeiPU-2 is the best mask for the image enhancement. Yi-Fei masks are shown in table (3.2) [18].

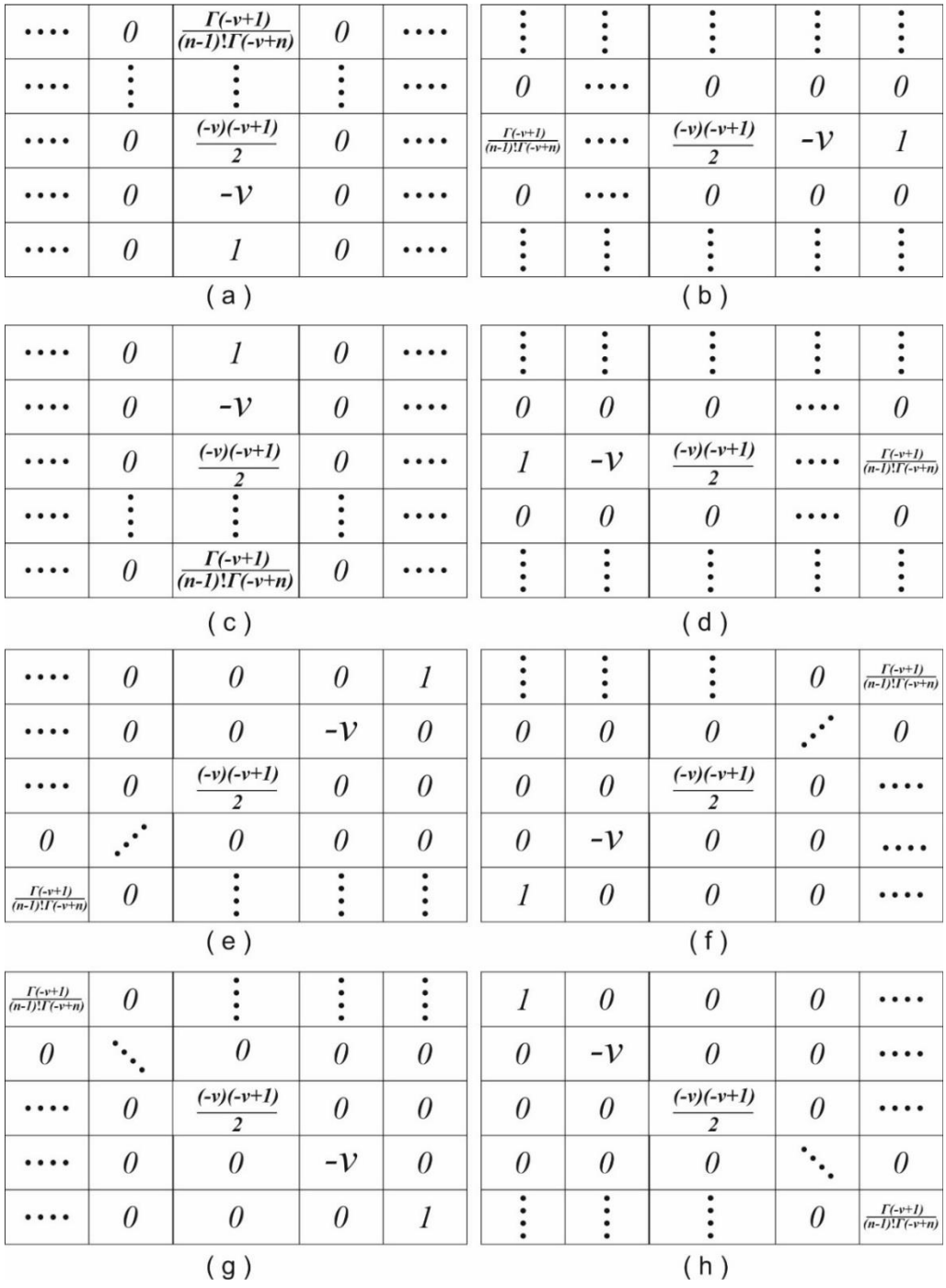


Fig 3.3: Fractional Differential mask in eight directions.

Table 3.2: Yi-Fei mask operators [18].

	YiFeiPU-1	***YiFeiPU-2***	***YiFeiPU-3 , v < 0***	***YiFeiPU-4 , v < 0***	***YiFeiPU-5,*** $\mathbf{0 \le v < 1}$	***YiFeiPU-6,*** $\mathbf{1 \le v < 2}$
C_{-1}		$\frac{v}{4}+\frac{v^2}{8}$				$\frac{1}{\Gamma(3-v)}$
C_0	1	$1-\frac{v^2}{2}-\frac{v^3}{8}$	$\frac{1}{\Gamma(-v)(-2v)}$	$\frac{1}{\Gamma(-v)(v^2-v)}$	$\frac{1}{\Gamma(2-v)}$	$\frac{2^{2-v}-3}{\Gamma(3-v)}$
C_1	$-v$	$-\frac{5v}{4}+\frac{v^3}{16}+\frac{v^4}{16}$	$\frac{2^{-v}}{\Gamma(-v)(-2v)}$	$\frac{2^{1-v}-2}{\Gamma(-v)(v^2-v)}$	$\frac{2^{1-v}-2}{\Gamma(2-v)}$	$\frac{3-3.2^{2-v}+3^{2-v}}{\Gamma(3-v)}$
⋮	⋮	⋮	⋮	⋮	⋮	⋮
C_n	$\frac{\Gamma(n-v)}{(n)!\,\Gamma(-v)}$	$\frac{\Gamma(n-v-1)}{(n-1)!\,\Gamma(-v)}\cdot\left(-\frac{v}{4}+\frac{v^2}{8}\right)$	$\frac{n^{-v}-(n-1)^{-v}}{\Gamma(-v)(-2v)}$	$\frac{(1-v)n^{-v}-n^{1-v}+(n-1)^{1-v}}{\Gamma(-v)(v^2-v)}$	$\frac{(1-v)n^{-v}-n^{1-v}(n-1)^{1-v}}{\Gamma(2-v)}$	$\frac{(2-3v+v^2)n^{-v}-(2-v)n^{1-v}+n^{2-v}-(n-1)^{2-v}}{\Gamma(3-v)}$

Chapter Four

Deblurring Based 2D Fractional Order Differentiation

4.1 Introduction

The software implementation of the proposed deblurring algorithm will be presented in the current chapter. MATLAB package has been used as a tool to implement software of the blind deblurring system. Quantitative and qualitative metrics have been used to qualify the proposed deblurring algorithm.

4.2 Problem Definition

The proposed deblurring algorithm involves using 2D-FOD operator instead of the point spread function in the enhanced version of Van Cittert algorithm [28]. The proposed algorithm is non-iterative. It is given in equation 4.1.

$$f = g + K(g - d^{v}(g)) \tag{4.1}$$

Where,

g : blurred image.
f : deblurred image.
K : gain.
d^{v}: 2D-fractional order differentiation.

The enhanced version of Van Cittert algorithm in [28] is an iterative non-blind deblurring algorithm in which a prior information about PSF must be known. In equation 4.1, there is no need for prior information about PSF. Genetic Algorithm (GA) as an optimization technique will be used to search

for the optimal values of K and v that give best quality of the deblurred image for out of focus blur.

4.3 2D-FOD Operator

The eight symmetric directions of the 2D-FOD mask have anti-rotation capability. 2D-FOD masks which are respectively on the directions of positive x-coordinate, left upward diagonal, positive y-coordinate, right upward diagonal, negative x-coordinate, left downward diagonal, negative y-coordinate and right downward diagonal are implemented in a single mask as shown in figure (4.1). YiFei-PU-2 mask is considered the best performance fractional order differential mask for texture enhancement. For 5x5 YiFei-PU-2 2D-FOD Mask is given in equation (4.2) [2].

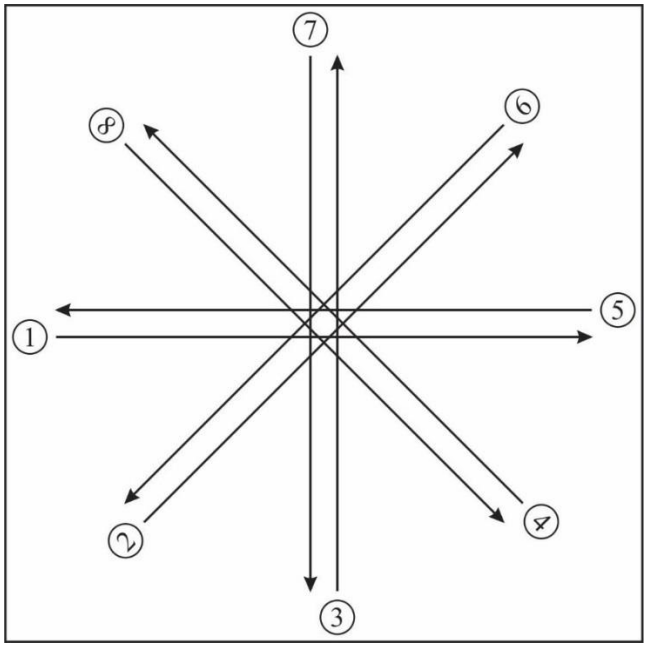


Fig. (4.1): Eight symmetric directions.

$$\begin{bmatrix} c_3+c_{-1} & 0 & c_3+c_{-1} & 0 & c_3+c_{-1} \\ 0 & c_2+c_0 & c_2+c_0 & c_2+c_0 & 0 \\ c_{-1}+c_3 & c_0+c_2 & 4(c_1+c_1) & c_2+c_0 & c_3+c_{-1} \\ 0 & c_0+c_2 & c_0+c_2 & c_0+c_2 & 0 \\ c_{-1}+c_3 & 0 & c_{-1}+c_3 & 0 & c_{-1}+c_3 \end{bmatrix} \quad (4.2)$$

Where:

$$c_{-1} = \frac{v}{4} + \frac{v^2}{8}\,, c_0 = 1 - \frac{v^2}{2} - \frac{v^3}{8}\,, c_1 = -\frac{5v}{4} + \frac{v^3}{16} + \frac{v^4}{16},$$

$$c_2 = \frac{\Gamma(1-v)}{(1)!\Gamma(-v)} \cdot \left(-\frac{v}{4} + \frac{v^2}{8}\right), and\ \ c_3 = \frac{\Gamma(2-v)}{(2)!\Gamma(-v)} \cdot \left(-\frac{v}{4} + \frac{v^2}{8}\right)$$

,

4.4 The Genetic algorithm

The Genetic Algorithms (GAs) is an optimization technique based on random search to obtain the best solution in a complex search space. In GA, the natural evolution is modeled to process population individuals in several generations to enhance the objective function. The individuals which represent an elected solution to the studied issue, usually known as chromosome. The chromosome is represented as a bit string [59].

GA has two primary issues, the first is called coding, it's how to code the issue. There are two common ways of coding, real or binary coding. The GA requires a collection of certain solutions called initial populations which are randomly selected. The second issue is called the fitness evaluation (objective function) it's how to qualify each individual chromosome (string). This step depends on nature of the issue. It is either rule-based procedure, mathematical equation or combination of the both [60].

GA has three main operators to create a new generation of chromosomes (individuals). These are reproduction (selection), crossover, and mutation. Reproduction is performed using one of the selection algorithms (Tournament, Roulette Wheel, ..etc.). Crossover is the process of

producing a child (offspring) from parent individuals. Mutation mimics biological mutation. Mutation changes gene value or genes value in the chromosome. It is used to avoid trapping the solution in local minima [61].

The following main steps illustrate the genetic algorithm:

Step 1: Initial population creation.

Step 2: Fitness calculation for each string.

Step 3: While (does not exceed maximum number of generation) OR (an acceptable solution is not found)

- Next generation reproduction using modified roulette selection.
- Create new offspring using crossover between parents.
- Perform mutation with certain probability.
- Fitness calculation for each offspring.

Step-4: End while.

4.4.1 Genetic-based 2D-FOD Operator

The undertaken optimization problem is to find the gain (K) and order (v) of the 2D-FOD of equation (4.1) which assure the objective function. The suggested objective function is to deblur the degraded image is based a simple fuzzy set of entropy and PSNR. Each GA solution has to minimize the following objective function:

$$Fitness = \frac{a_0}{Entropy} + \frac{a_1}{PSNR} \tag{4.3}$$

Where:

a_0: is entropy weight, between$(0\ and\ 1)$.

a_1: is PSNR weight ($a_1 = 1 - a_0$).

$$PSNR = 10 \log\left(\frac{255^2}{\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(g_{i,j} - I_{i,j}\right)^2}\right) \tag{4.4 [62]}$$

$$Entropy = -\sum_x \sum_y f(x,y) \log\big(f(x,y)\big) \tag{4.5 [62]}$$

$g_{i,j}$: blurred image.

$I_{i,j}$: original image.

$f(x,y)$: restored image.

The deblurring algorithm parameters is coded in a binary string as shown in figure (4.2).

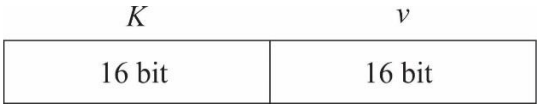
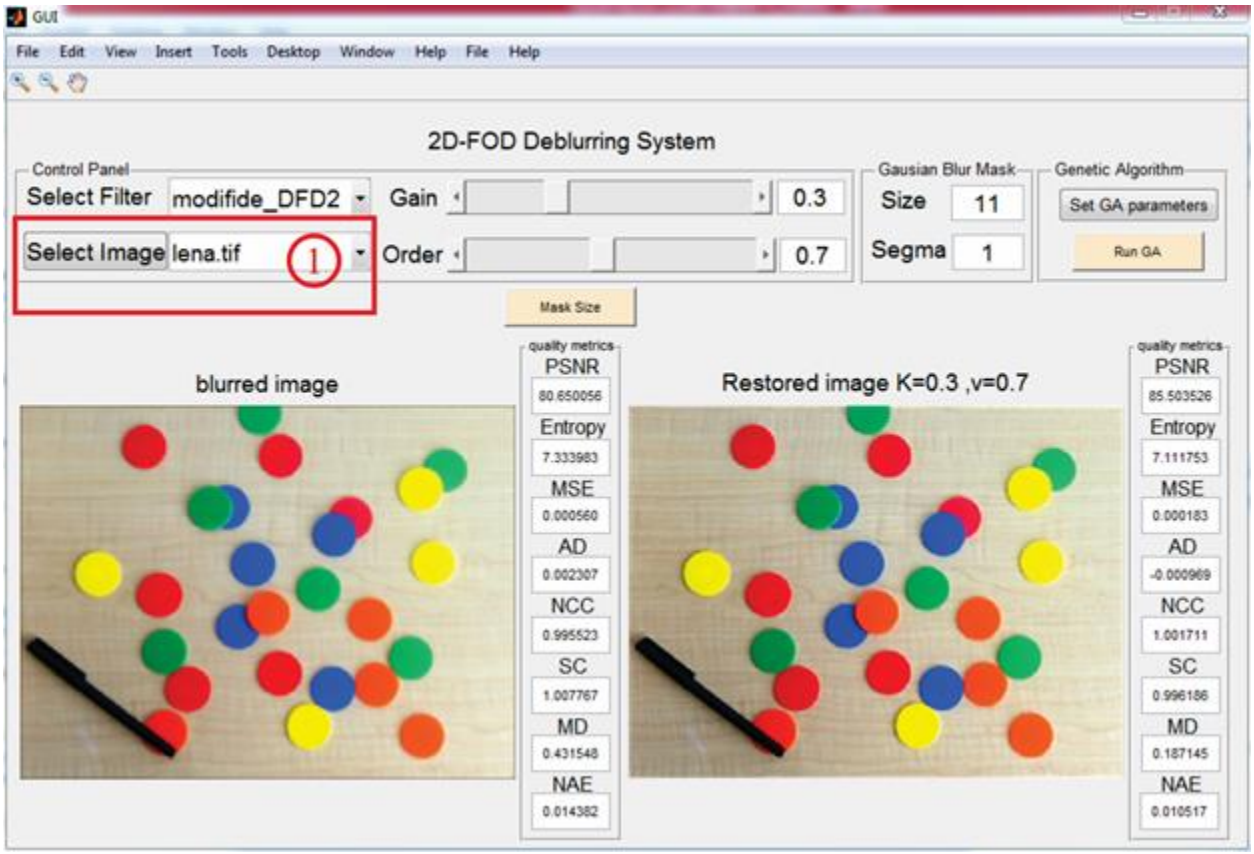


Fig. (4.2): FOD Deblurring system parameter string codding.

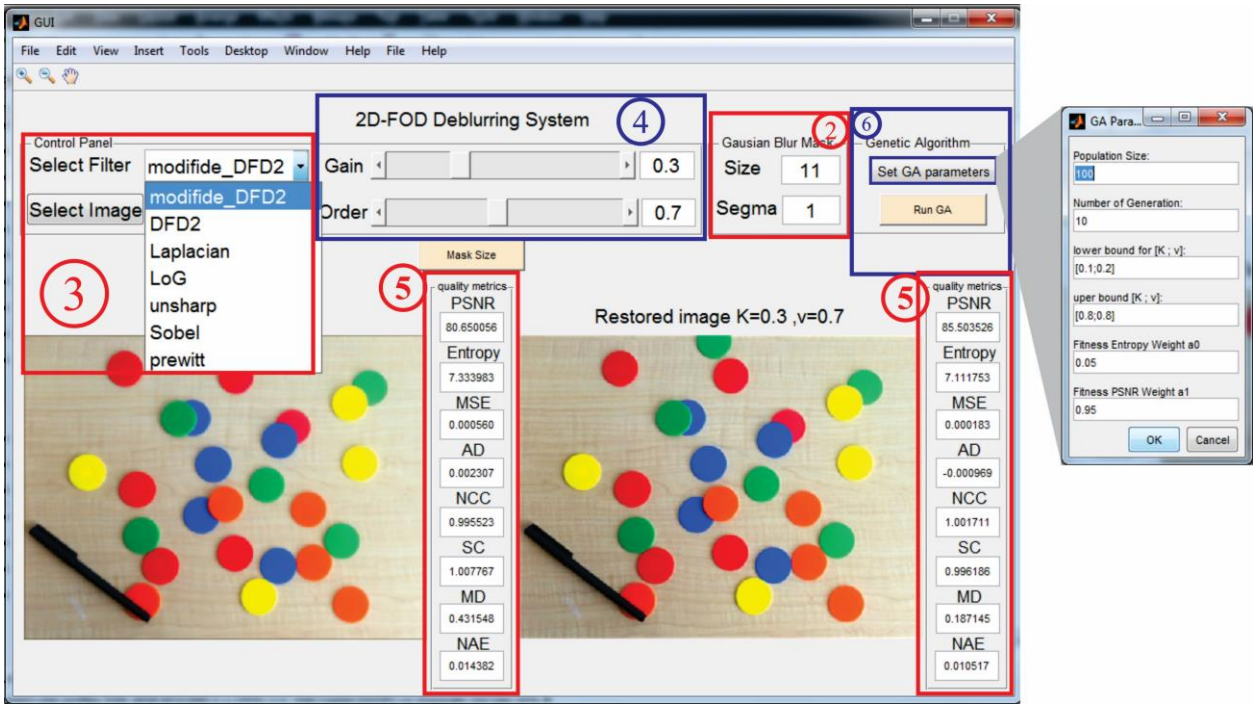
4.5 Software Implementation

The graphical user interface (GUI) of the 2D-FOD deblurring system software which has been implemented in Matlab is shown in figure (4.3). The source code in given in appendix-A. It consists of the following items:-

1- Select an image using browse file.
2- Select mask size and sigma from Gaussian blur mask panel. It is used to simulate out-of-focus blur.
3- Select filter (modified_DFD2, DFD2, Laplacian, LoG, unsharp, sobel, and prewitt) from control panel to deblur degraded image version.
4- Modified_DFD2 filter will be manually pass the gain (K) and order (v) to apply equation 4.1, from gain and order slide bar.



(a)



(b)

Fig (4.3): 2D-FOD Deblurring system GUI.

5- The following metrics will be calculated for the restored image [62].

a. Peak Signal to Noise Ratio (PSNR in dB):

$$PSNR = 10\log\left(\frac{255^2}{\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(g_{i,j} - I_{i,j}\right)^2}\right) \quad (4.6)$$

b. Entropy:

$$Entropy = -\sum_{x}\sum_{y} f(x,y)\log\left(f(x,y)\right) \quad (4.7)$$

c. Mean Square Error (MSE):

$$MSE = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(g_{i,j} - I_{i,j}\right)^2 \quad (4.8)$$

d. Normalized Cross-Correlation (NCC):

$$NCC = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} g_{i,j} \cdot I_{i,j}}{\sum_{i=1}^{M}\sum_{j=1}^{N} g_{i,j}^2} \quad (4.9)$$

e. Average Difference (AD):

$$AD = \sum_{i=1}^{M}\sum_{j=1}^{N}\frac{\left(g_{i,j} - I_{i,j}\right)}{MN} \quad (4.10)$$

f. Structural Content (SC):

$$SC = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} g_{i,j}^2}{\sum_{i=1}^{M}\sum_{j=1}^{N} I_{i,j}^2} \quad (4.11)$$

g. Maximum Difference (MD):

$$MD = Max\left(\left|g_{i,j} - I_{i,j}\right|\right) \quad (4.12)$$

h. Normalized Absolute Error (NAE):

$$NAE = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}\left|g_{i,j} - I_{i,j}\right|}{\sum_{i=1}^{M}\sum_{j=1}^{N}\left|g_{i,j}\right|} \quad (4.13)$$

Where:

$g_{i,j}$ is blurred image.

$I_{i,j}$ is original image.

6- Now to find the best gain (k) and order (v) for proposed algorithm, GA will be used. The fitness function depends on PSNR and Entropy in equation (4.3).

4.6 Results Analysis

Three parameters of equation (4.1) must be determined $(v, k, and\ mask\ size)$ in order to implement the proposed algorithm using an FPGA. Quantitative and qualitative approaches have been used to achieve the proper selection of the parameters.

4.6.1 Quantitative Analysis

Different metrics (equations 4.6 – 4.13) have been used to qualify the restored images. PSNR for various orders and gains at different mask sizes (3, 5, 7, and 9) is shown in figure (4.4), whereas Entropy is shown in figure (4.5). The proposed 2D-FOD algorithm has been compared with standerd integar order differentiation (sobel, prewitt, and Laplacian) as illustrated in figure (4.6) and figure (4.7). The performance of the current approach for debluuring images degraded by out of focuse blur is better than the traditional filters (for more detials see appendix-B). According to these empirical results, GA has been used to search the optimal gain and order using fitness of equation (4.3) at certain range of gains and orders.

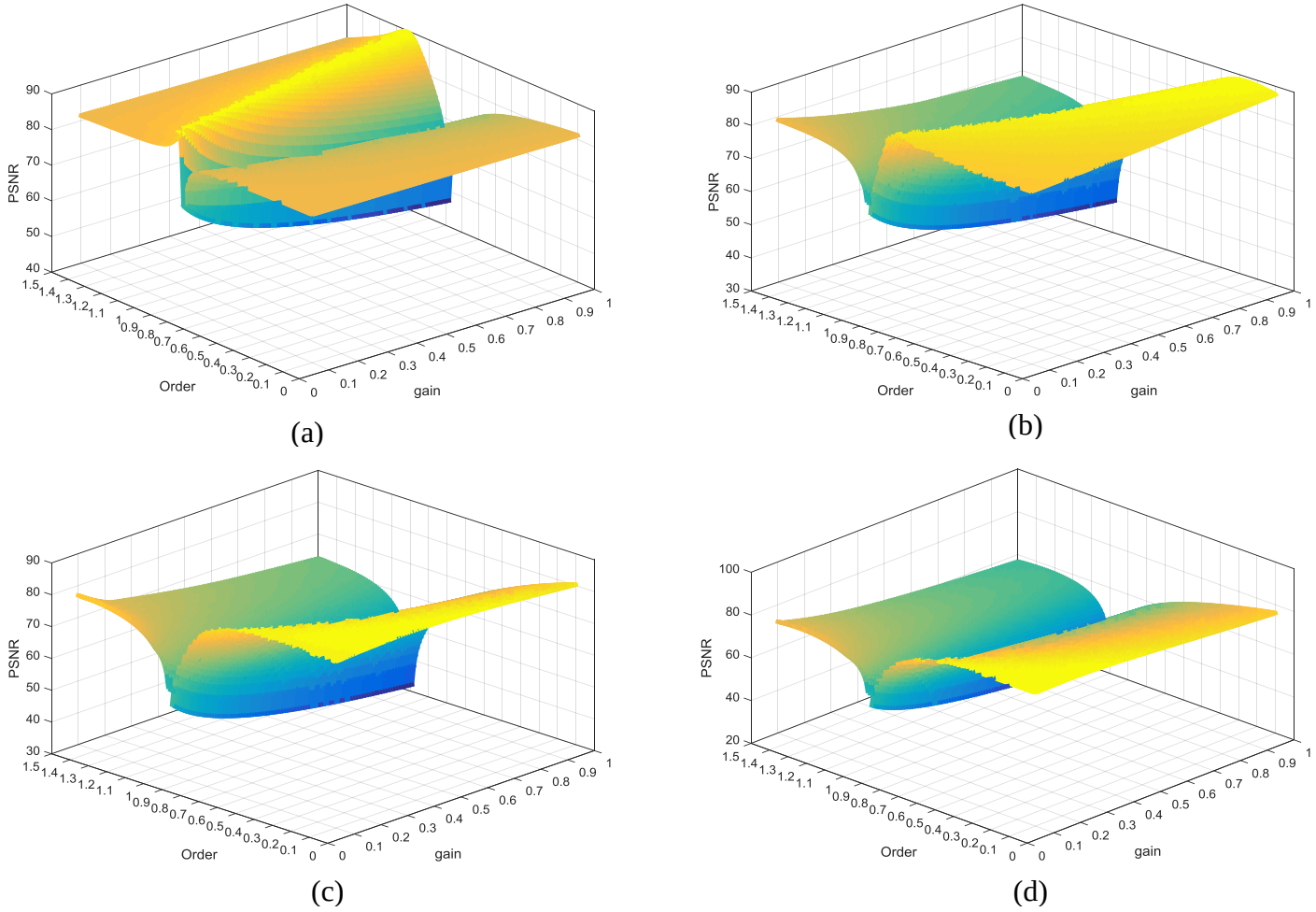


Fig (4.4): PSNR for cat image (a).Mask Size 3x3, (b). Mask Size 5x5, (c). Mask Size 7x7 and (d). Mask Size 9x9.

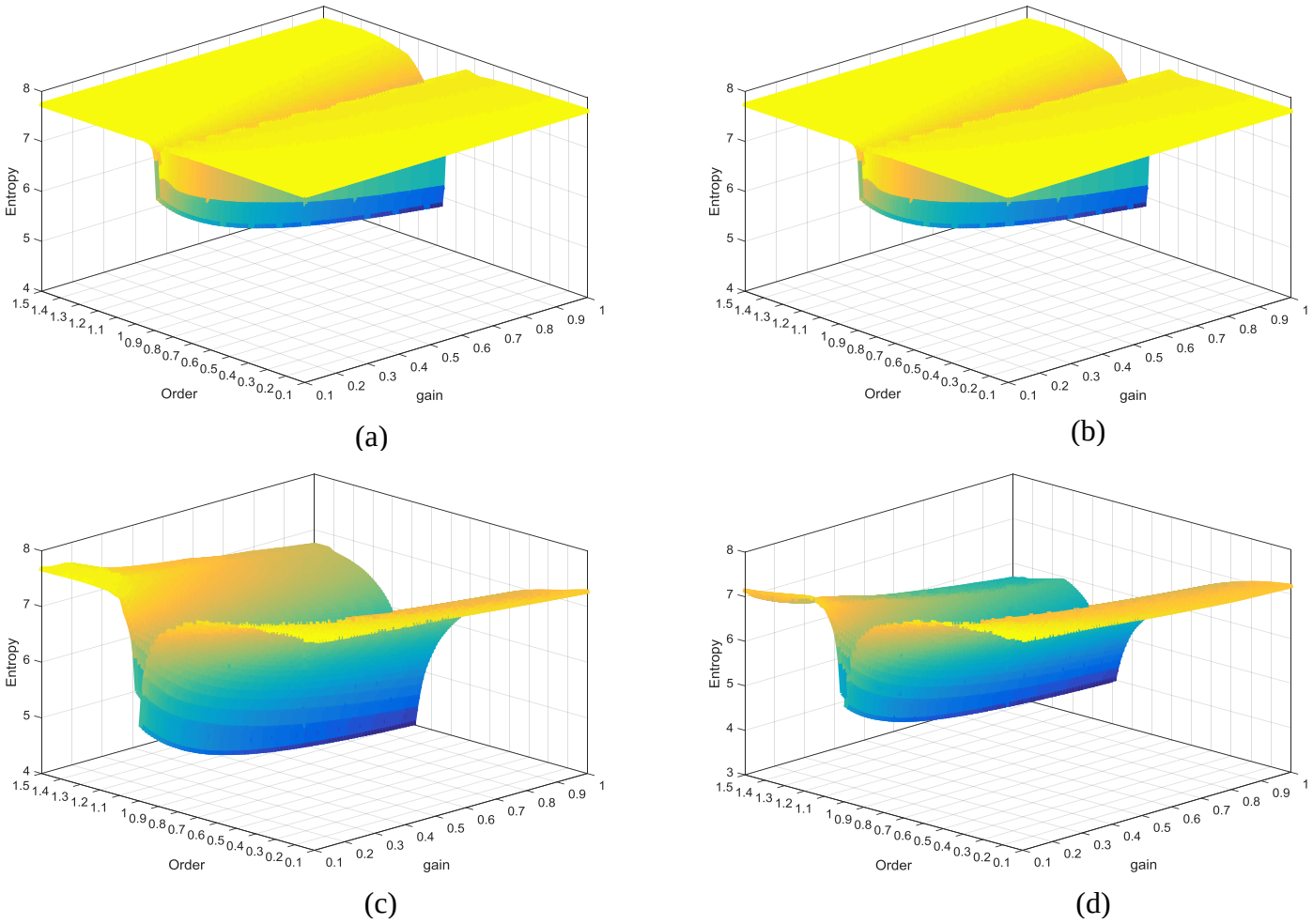


Fig (4.5): Entropy for cat image (a).Mask Size 3x3, (b). Mask Size 5x5, (c). Mask Size 7x7 and (d). Mask Size 9x9.

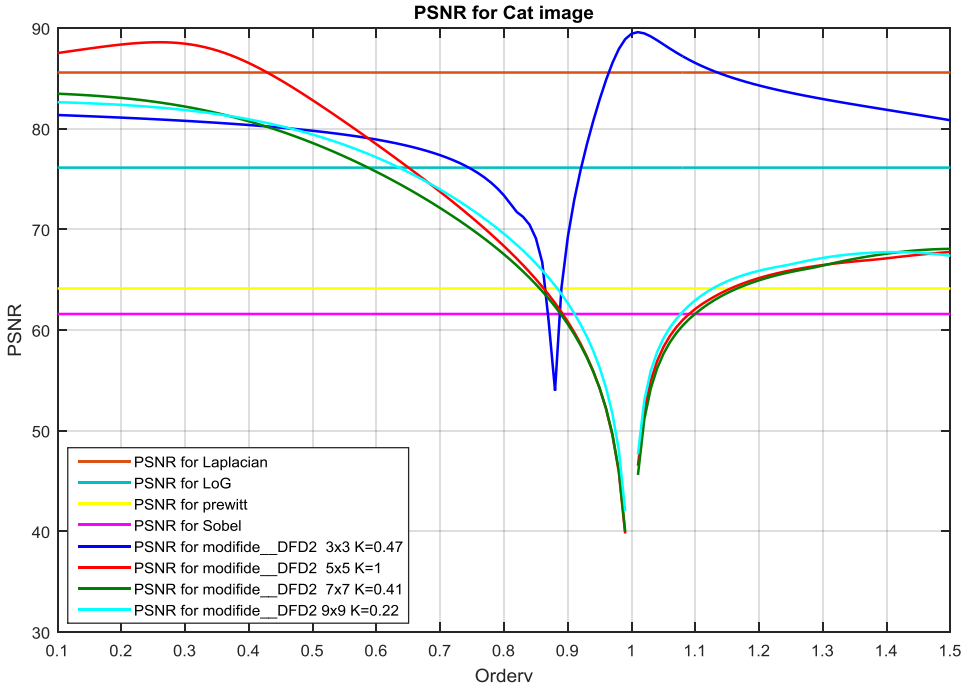


Fig (4.6): PSNR for cat image.

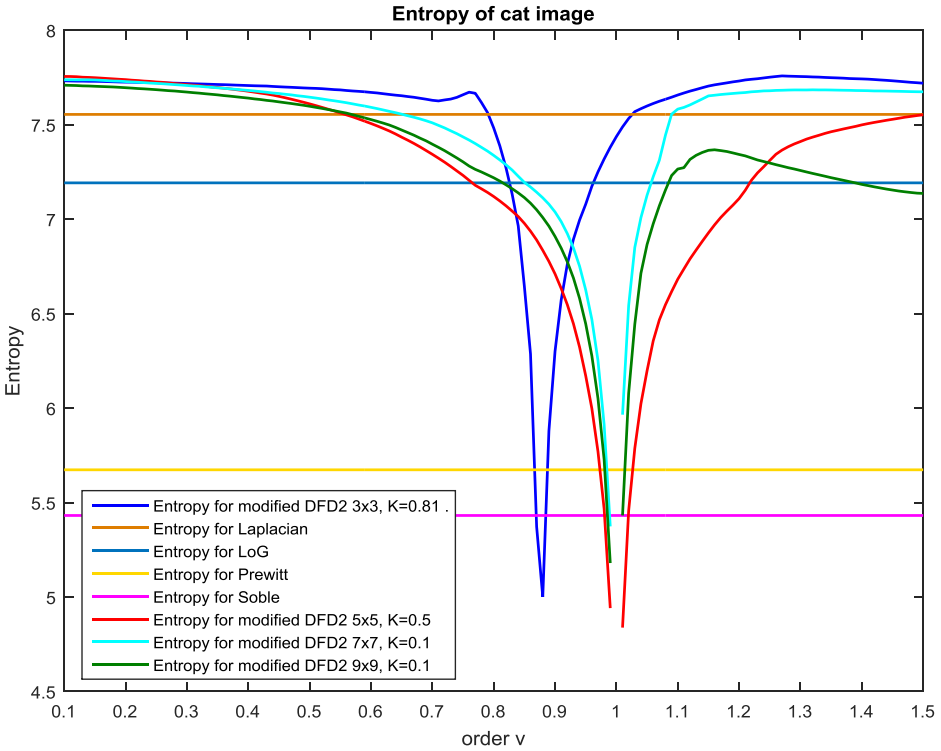


Fig (4.7): Entropy for cat image.

Table 4.1: Cat video frames K=0.5, v=0.6

Frame No.	Input Frame	Output Frame
1		
20		
40		
60		

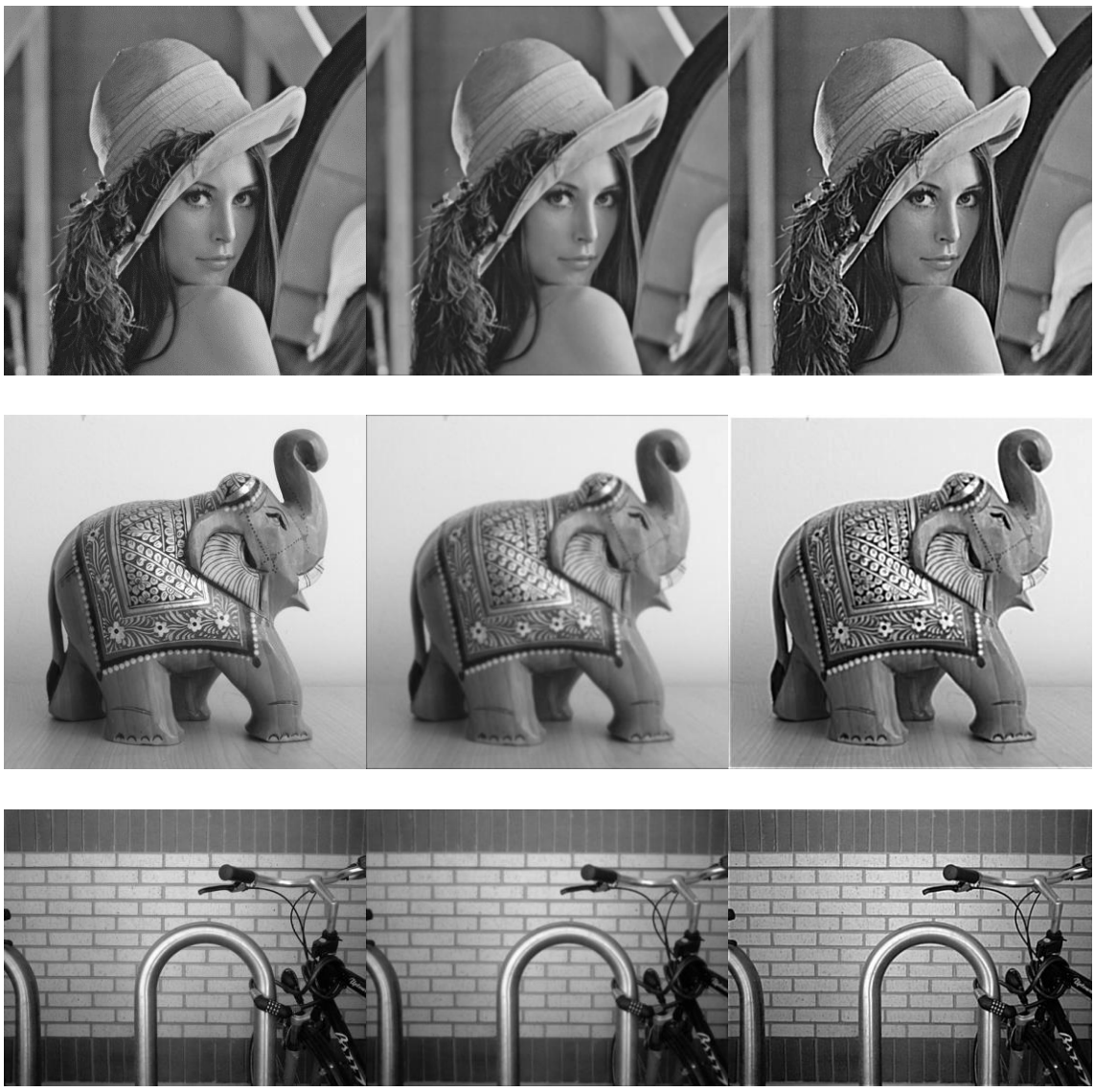
90		
100		
120		
130		
160		

180		
190		
200		
210		
220		

4.6.2 Qualitative Analysis

An online questionnaire survey using google forms (available at https://goo.gl/forms/wAdhoGMeSfHPsUDy1) has been used for visual qualification of the restored images. Figure (4.8) shows the deblurring image was degraded by out of focus blue with sigma =1, whereas figure (4.9) Sega =2 see appendix-C. Appendix-D shows samples of the forms of questionnaire survey. The questionnaire consists of two parts. The first compares between the restored image using 2D-FOD algorithm and the degraded one. Whereas the other compares between the restored using Laplacian filter and the proposed algorithm (refer to figure (4.6) and (4.7)). Table 4.2 presents the summary of the questionnaire survey of 125 voters.

Original image Blurred image Restored image



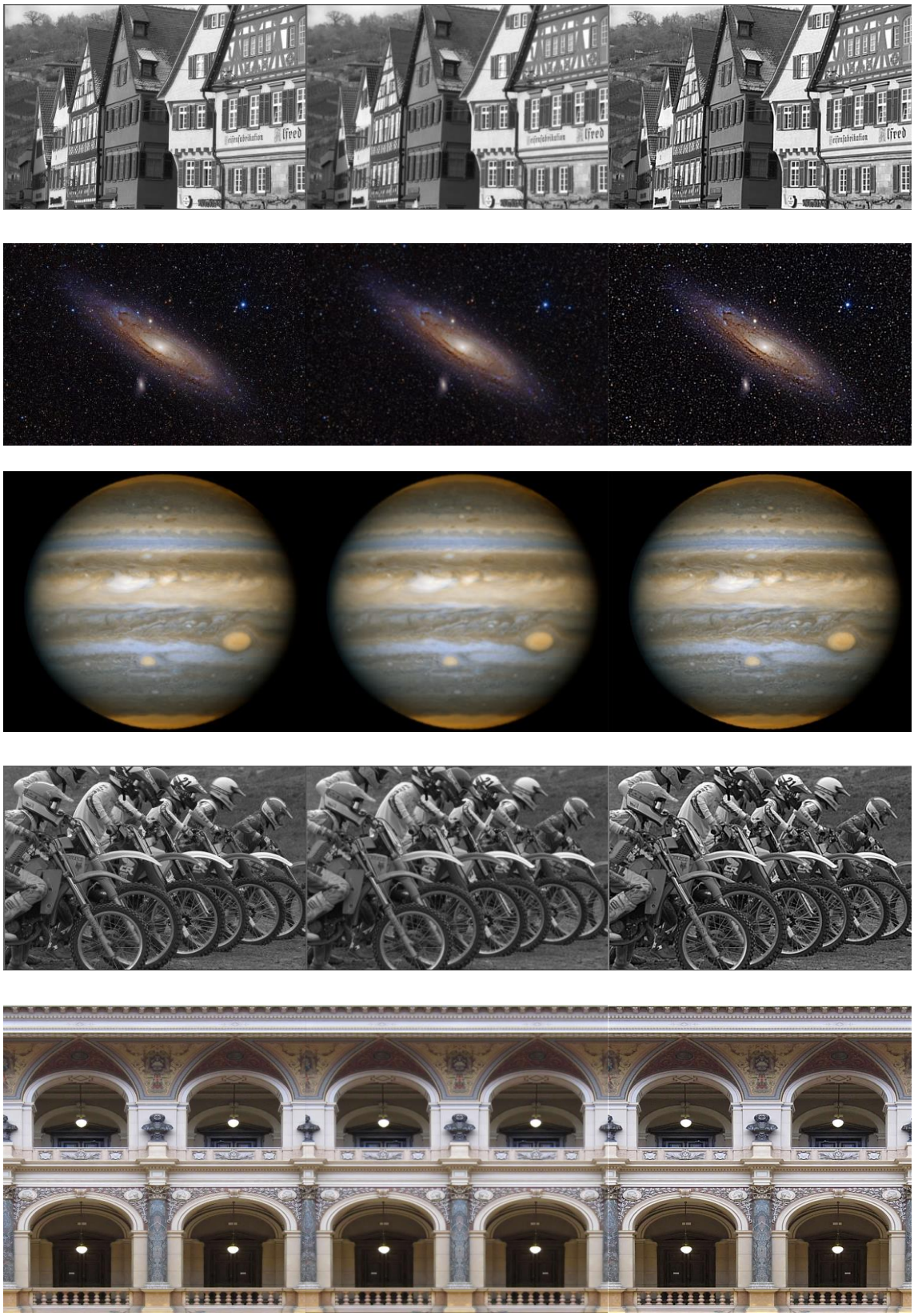


Fig (4.8): Gaussian blur using sigma $\sigma = 1$ and $11x11$ kernel size Deblurring using modified DFD2 K=0.5 and v=0.75.

Original image Blurred image Restored image

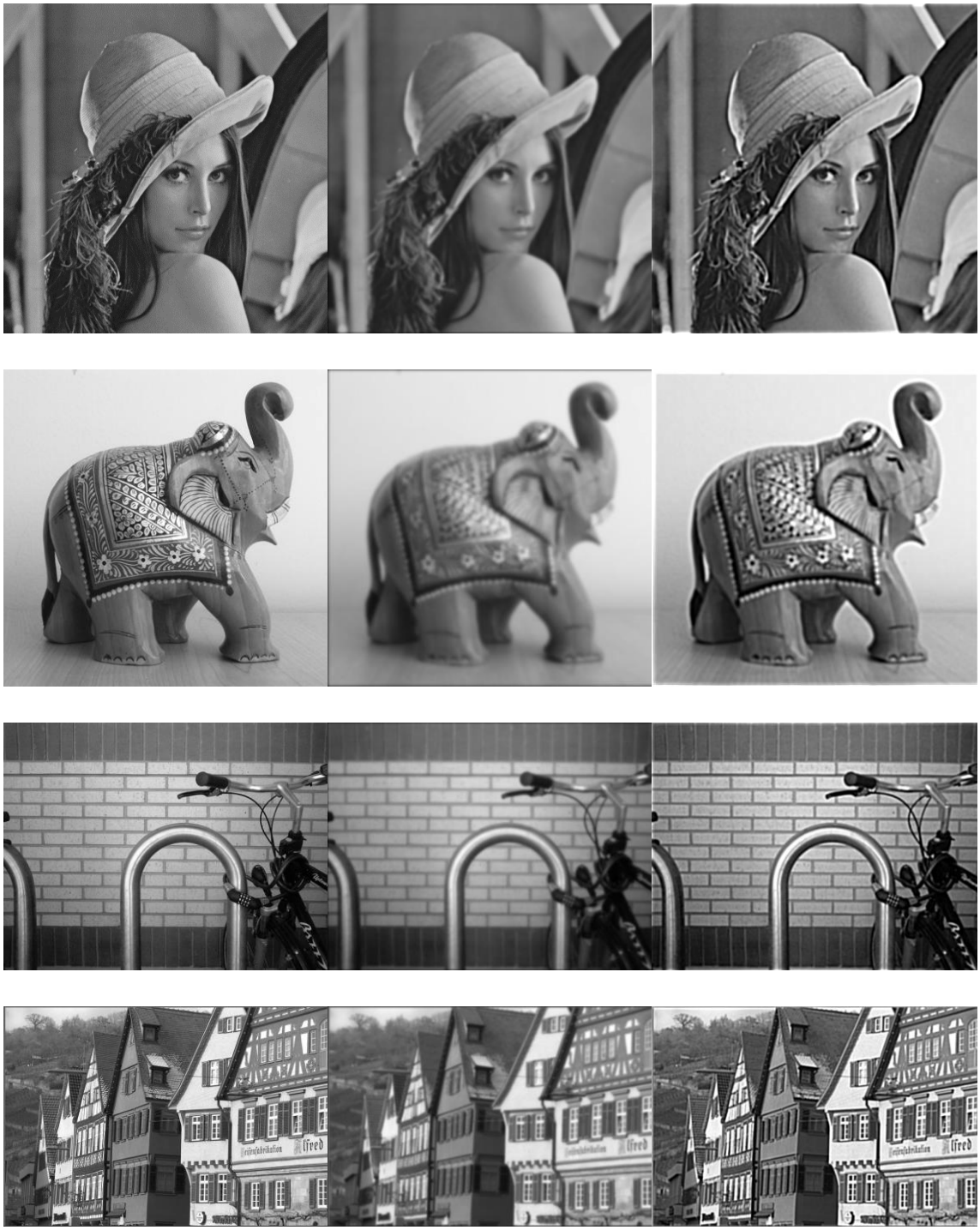


Fig (4.9): Gaussian blur using sigma $\sigma = 2$ and $11x11$ kernel size

Deblurring using modified DFD2 K=0.8 and v=0.88.

Table 4.2: Summary of the questionnaire survey.

	First questionnaire forms		second questionnaire forms	
Images	blurred	restored	Laplacian	Modified FOD2
Elephant	0%	100%	4%	96%
Bikes gray	1.6%	98.4%	8.8%	91.2%
Calculator	8.8%	91.2%	-	-
Houses	1.6%	98.4%	11.2%	88.8%
House	2.4%	97.6%	8%	92%
Parrots	2.4%	97.6%	15.2%	84.8%
Colored Chips	0.8%	99.2%	11.2%	88.8%
Flower	9.6%	90.4%	22.4%	77.6%
Palace	3.2%	96.8%	14.4%	85.6%
Forest	0.8%	99.2%	14.4%	85.6%
Peacock	0.8%	99.2%	10.4%	89.6%
Valley	0.8%	92.2%	10.4%	89.6%
Electronic Kit	0%	100%	8%	92%
Parthenon	0%	100%	11.2%	88.8 %
Cloth	0.8%	99.2%	12.8%	87.2%

Chapter Five

FPGA Implementation of 2D-FOD

5.1 Introduction

The FPGA implementation of the proposed deblurring algorithm will be presented in the current chapter. The algorithm has been implemented on Xilinx Spartan 6 (Xc6slx16). The hardware Systems can be designed using high-level Simulink graphical environment of Xilinx system generator (XSG). XSG provides graphical modules that expand the Hardware description Language (HDL). The Simulink enables the design abstraction using XSG subsystems and blocks which specifically reduces the necessary time for hardware implementation. Also, Simulink provides FPGA-in-the-loop verification and simulation, which is known as hardware cosimulation. This methodology provides easier hardware verification and implementation compared to HDL based approach. The Simulink simulation and hardware-in-the loop approach presents a far more cost efficient solution than other methodologies [3].

5.2 2D-FOD Deblurring System

The suggested Deblurring system using 2D-FOD (refer to equation 4.1) is shown in figure 5.1. The 2D-FOD system model is to be implemented in FPGA using two design methodologies. The first using DSP System Generator; and the second using Matlab HDL Verifier and HDL Coder, FPGA in the Loop (FIL).

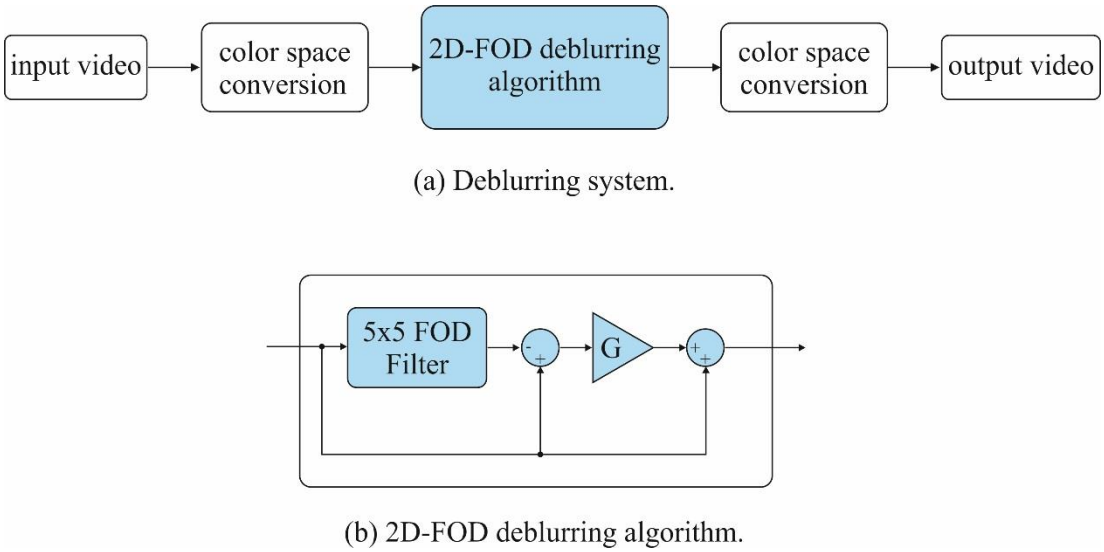


Fig 5.1: 2D-FOD deblurring system.

5.3 FPGA Implementation Design Methodologies

In MATLAB Simulink, there is a development platform that encapsulates the design of the system, system modeling, system simulation, code generation of the system, and system implementation. There are two design approaches. One uses Simulink HDL Coder. The other approach uses Xilinx System Generator. Table 5.1 illustrates the benefits and main features of System Generator and HDL Coder. Each approach gives an effective hardware design flow.

Table 5.1: System Generator and HDL Coder benefits and main features.

Feature	HDL Coder	System Generator	Benefit
MATLAB to HDL	X		Rapidly prototype algorithmic MATLAB
Floating-to Fixed-Point Conversion	X		Shorten design cycle
Design exploration	X		Rapidly explore hardware solution space
Software and Hardware code generation	X		Partition algorithms between processor and hardware
Access to Simulink block library	X		Rapidly assemble system model to hardware
Support for native Simulink block	X		Easily migrate from system model to hardware
Automatic test generation	X		Verify hardware against system model
Transaction Level Model (TLM) component generation	X		Support system level modeling
Readable, traceable HDL code	X		Streamline standards compliance and reporting
Access Xilinx IP in Simulink		X	Generate implementations optimized for Xilinx targets
Hardware co-simulation		X	Verify hardware implementations on Xilinx development boards
Analog data acquisition		X	Verify algorithms to real world analog data
Hardware deployment		X	Deploy design in hardware without FPGA design experience

5.3.1 Xilinx System Generator Based Design

The MATLAB R2011b with Xilinx System Generator for DSP and Integrated Software Environment (ISE14.2) are used as illustrated in figure 5.2.

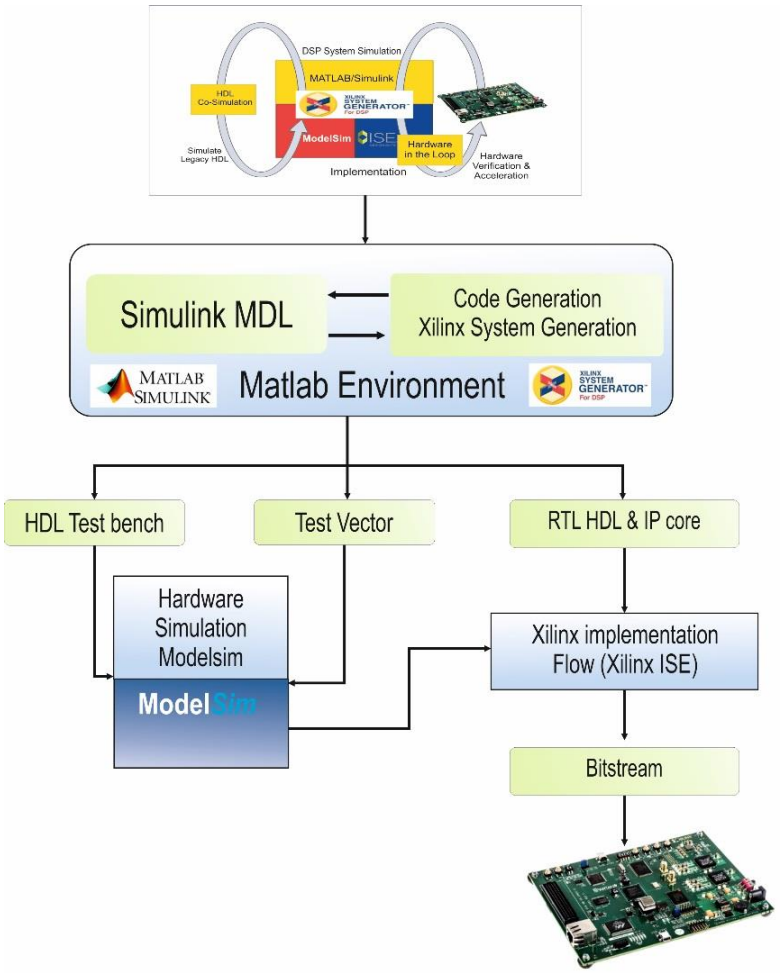


Fig 5.2: Design Methodology with Xilinx System Generator.

ISE is a powerful design environment that is working in the background when implementing System Generator blocks. The ISE environment consists of a set of program modules, written in HDL, that are utilized to create, capture, simulate and implement digital designs in a FPGA or Complex Programmable Logic Devices (CPLD). The netlist files are created in the synthesis process of the modules which serve as the input to the

implementation process. Then, the logic design is converted into a bitstream file that can be downloaded on the target FPGA device [3].

5.3.2 HDL Verifier and HDL Coder Based Design

HDL Verifier works with Simulink, MATLAB, HDL Coder and the supported FPGA development environment to prepare automatically the generated HDL Code for implementation in an FPGA. FIL simulation allows running a Simulink or MATLAB simulation with an FPGA board strictly synchronized with the software. This process allows getting a real world data into the design while accelerating the simulation with the speed of an FPGA as illustrated in figure 5.3.

FIL provides the communication channel for sending and receiving data between Simulink and the FPGA. This channel uses a Gigabit Ethernet connection. Because communication between Simulink and the FPGA is strictly synchronized, the FIL simulation provides a more dependable verification method [13].

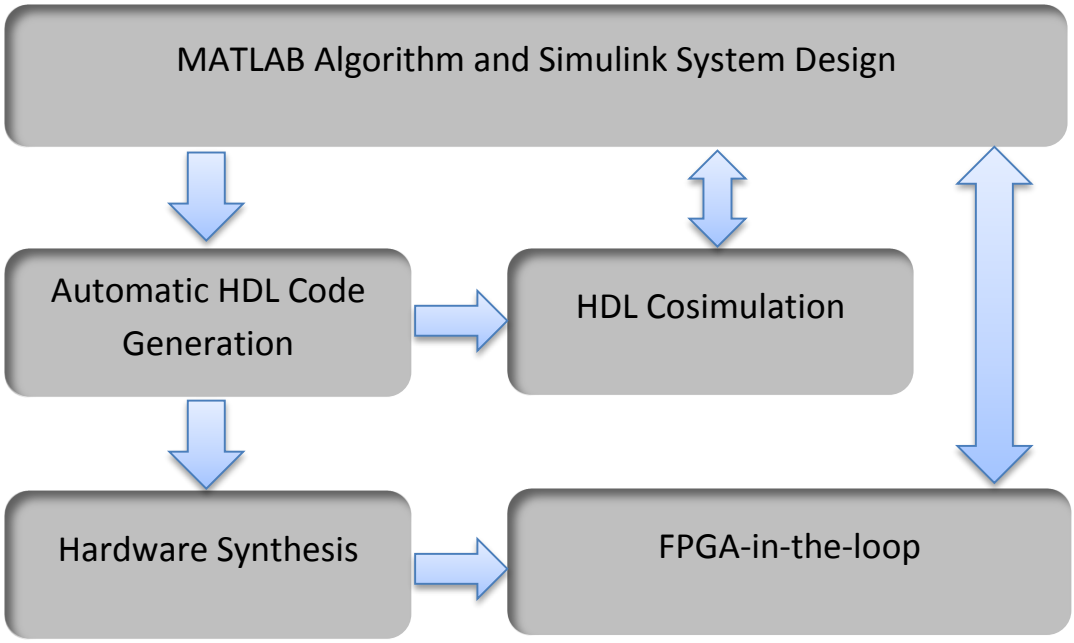


Fig 5.3 Design Methodology using FIL.

5.4 Hardware Implementation Using System Generator

The model of video deblurring system that implemented using System Generator is showing in figure 5.4.

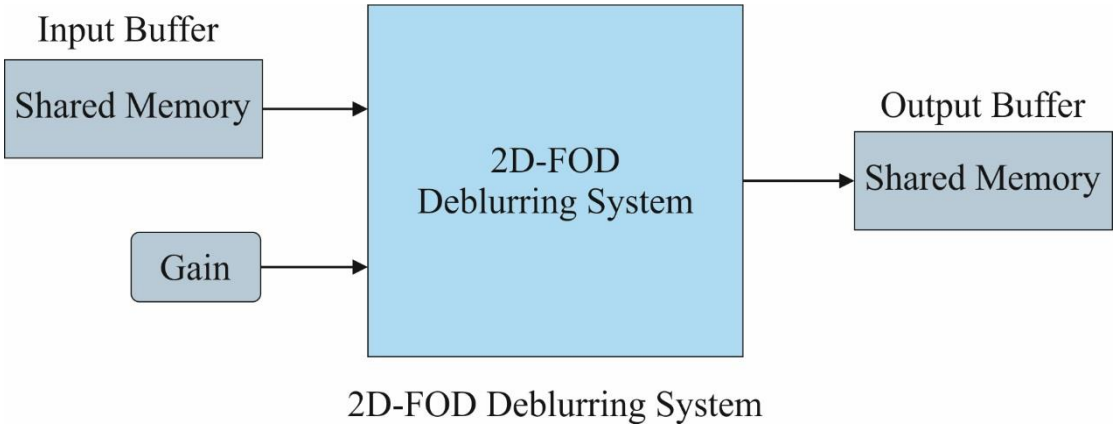


Fig 5.4: Hardware model of Deblurring System.

The shared memory enables of image processing tasks with specific memory requirements and high bandwidth to be simulated on FPGA. The data transactions between FPGA device and Simulink are carried out using burst transfers which provide a throughput for real time image processing applications depending on the type of cosimulation interface.

Figure 5.5 shows I/O buffering interface that enables data buffering and streaming using XSG data path during cosimulation. The design is composed of input and output subsystems to realize the buffer storage. The middle block of the figure is a placeholder for the proposed image processing algorithm. The buffering interface contains data valid ports which control the data flow [12].

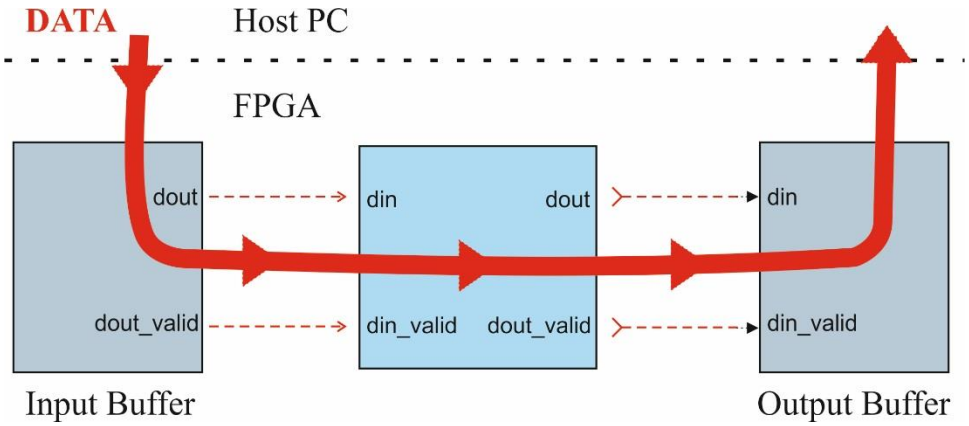


Fig 5.5: Operation of I/O buffering interface and Data Path.

5.4.1 Two Dimensional FOD Deblurring System

Refer to figure 5.1.b, the deblurring system consists of 2D-FOD filter mask, gain, subtracter and adder. The contents of the 5x5 filter mask is illustrated in figure 5.6. It contains a line buffer. The line buffer can be configured to accommodate certain frame size to process a complet frame during single simulation cycle. The coefficient memory stores the filter coefficients. The coefficients of the filter mask can be loaded dynamically during run time.

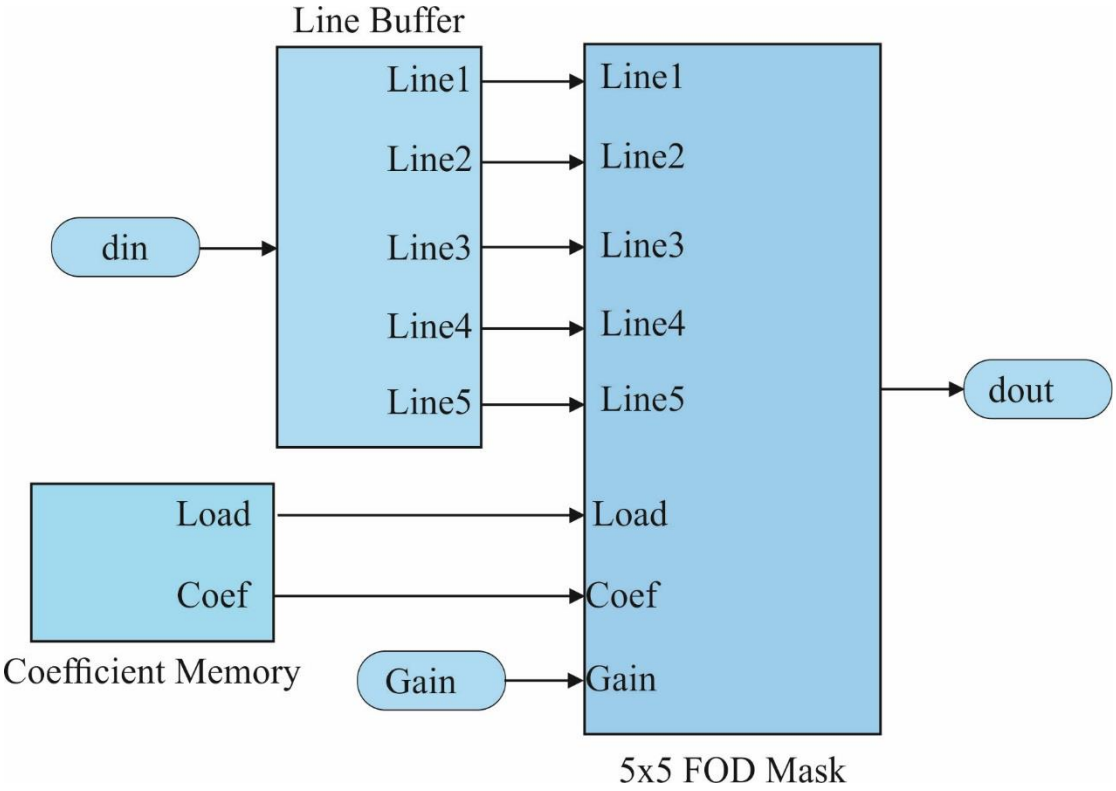


Fig 5.6: Two dimensional FOD.

XSG provides a high-throughput cosimulation with FPGA board over an Ethernet (Point-to-point) connection to eliminate the limitations caused by programming cables [12].

5.4.2 Compilation for Cosimulation

Before hardware cosimulation, the hardware design must be compiled. The compilation phase generates a hardware cosimulation block as illustrated in figure 5.7. The block includes information about FIFOs, registers, and shared memories.

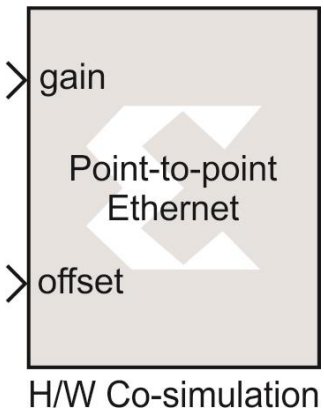


Fig 5.7: Hardware Cosimulation block

5.4.3 2D-FOD Deblurring System Test Bench

The deblurring system test bench consists of FPGA processing, video source, and matrix viewer (video viewer) models as illustrated in figure 5.8. Each frame (128x128 uint8) of the video stream is processed on the FPGA model. The FPGA processing model is implemented on Xilinx SP601 development board. Refer to figure 5.9, the operation of the deblurring processing model is expressed as follows:

a) The host PC locks the Shared Memory Write block from access by FPGA processing model. Then, the host PC writes a video frame into the shared memory and releases the lock.

b) The hardware cosimulation model locks the read and write shared memories. Then, the FPGA model processes the input frame and stores the result in the shared memory read block. Lastly, the FPGA model grants the shared memories for host PC access.

c) The host PC locks the Shared Memory Read block. Then, the PC reads the output frame and displays it on the matrix viewer.

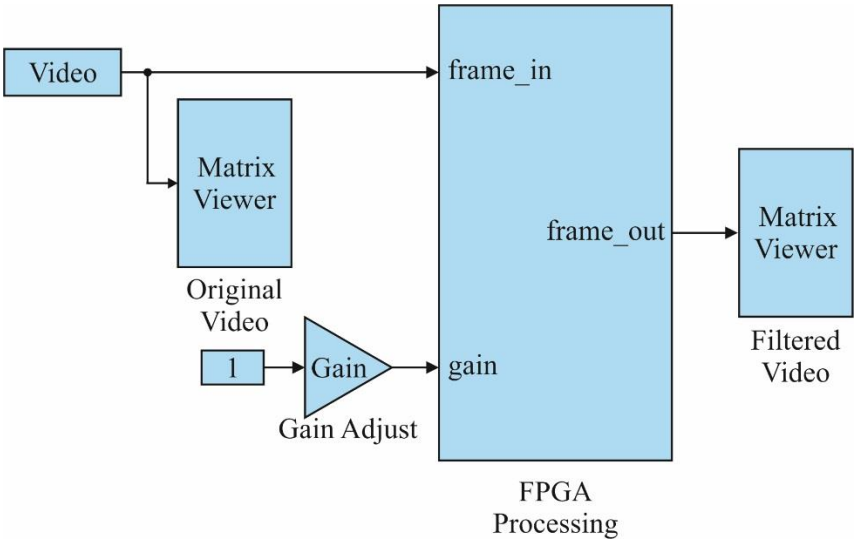


Fig 5.8: 2D-FOD Deblurring System Test Bench.

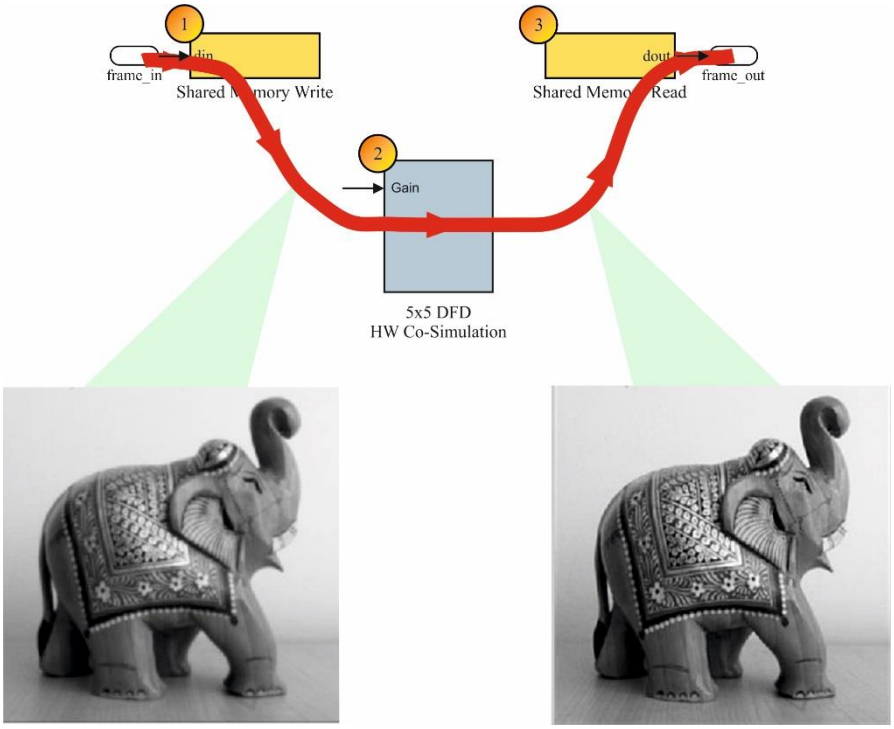


Fig 5.9: System Hardware Cosimulation.

5.4.4 Hardware Implementation Reports

1. Gray Image

The synthesis has been implemented on one plane (gray) image. The summery of utilization report is given in table 5.2.

Table 5.2 Gray Image Device Utilization Summary

Logic Utilization	Used	Available	Utilization
Number of Slice Registers	899	18224	4%
Number of Slice LUTs	482	9112	5%
Number of fully used LUT-FF pairs	351	1030	34%
Number of bonded IOBs	133	232	57%
Number of BUFG/BUFGCTRLs	1	16	6%
Number of DSP48A1s	12	32	37%

2. RGB Image

The synthesis has been implemented on three planes (RGB) image. The summery of utilization report is given in table 5.3.

Table 5.3 Three planes (RGB) image Device Utilization Summary**.**

Logic Utilization	Used	Available	Utilization
Number of Slice Registers	1453	18224	7%
Number of Slice LUTs	798	9112	8%
Number of fully used LUT-FF pairs	573	1678	34%
Number of bonded IOBs	240	232	103%
Number of BUFG/BUFGCTRLs	1	16	6%

3. YCbCr Image

On YCbCr color space the synthesis has been implemented on one plane (Y plane) only. The summery of utilization report is shown in table 5.4.

Table 5.4 YCbCr Device Utilization Summary.

Logic Utilization	Used	Available	Utilization
Number of Slice Registers	553	18224	3%
Number of Slice LUTs	320	9112	3%
Number of fully used LUT-FF pairs	244	629	38%
Number of bonded IOBs	148	232	63%
Number of BUFG/BUFGCTRLs	1	16	6%
Number of DSP48A1s	7	32	21%

5.4.5 Hardware Minimization

The hardware required to implement the proposed deblurring algorithm in the FPGA can be minimized at high level abstraction by simplifying and reordering equation 4.1. In spatial domain, for mask of size 3x3 for example, the equation 4.1 can be written as:

$$\begin{aligned} f(x,y) = g(x,y) \\ &+ K\big[\, g(x,y) - [w_{(-1,-1)}g(x-1,y-1) \\ &+ w_{(0,-1)}g(x,y-1) + w_{(1,-1)}g(x+1,y-1) \\ &+ w_{(-1,0)}g(x-1,y) + w_{(0,0)}g(x,y) + w_{(1,0)}g(x+1,y) \\ &+ w_{(-1,1)}g(x-1,y+1) + w_{(0,1)}g(x,y+1) \\ &+ w_{(1,1)}g(x+1,y+1)\,]\big] \end{aligned} \tag{5.1}$$

$$\begin{aligned} f(x,y) \;=\; & -Kw_{(-1,-1)}\, g(x-1,y-1) - Kw_{(0,-1)}\, g(x,y-1) \\ & - Kw_{(1,-1)}\, g(x+1,y-1) - Kw_{(-1,0)}\, g(x-1,y) \\ & + \big(1 + K - Kw_{(0,0)}\big)g(x,y) - Kw_{(1,0)}\, g(x+1,y) \\ & - Kw_{(-1,1)}\, g(x-1,y+1) - Kw_{(0,1)}\, g(x,y+1) \\ & - Kw_{(1,1)}\, g(x+1,y+1) \end{aligned} \tag{5.2}$$

Equation 5.2 can be rewritten in matrix notation as:

$$F = A \circledast G \qquad (5.3)$$

Where:

$$A = \begin{bmatrix} Kw_{(-1,-1)} & Kw_{(0,-1)} & Kw_{(1,-1)} \\ Kw_{(-1,0)} & 1 + K - Kw_{(0,0)} & Kw_{(1,0)} \\ Kw_{(-1,1)} & Kw_{(0,1)} & Kw_{(1,1)} \end{bmatrix} \qquad (5.4)$$

For 5x5 mask size kernel,

$$A = \begin{bmatrix} Kw_{(-2,-2)} & Kw_{(-1,-2)} & Kw_{(0,-2)} & Kw_{(1,-2)} & Kw_{(2,-2)} \\ Kw_{(-2,-1)} & Kw_{(-1,-1)} & Kw_{(0,-1)} & Kw_{(1,-1)} & Kw_{(2,-1)} \\ Kw_{(-2,0)} & Kw_{(-1,0)} & 1 + K - Kw_{(0,0)} & Kw_{(1,0)} & Kw_{(2,0)} \\ Kw_{(-2,1)} & Kw_{(-1,1)} & Kw_{(0,1)} & Kw_{(1,1)} & Kw_{(2,1)} \\ Kw_{(-2,2)} & Kw_{(-1,2)} & Kw_{(0,2)} & Kw_{(1,2)} & Kw_{(2,2)} \end{bmatrix} (5.5)$$

The system hardware shown in figure 5.1.b will be as shown in figure 5.10 after the minimization.

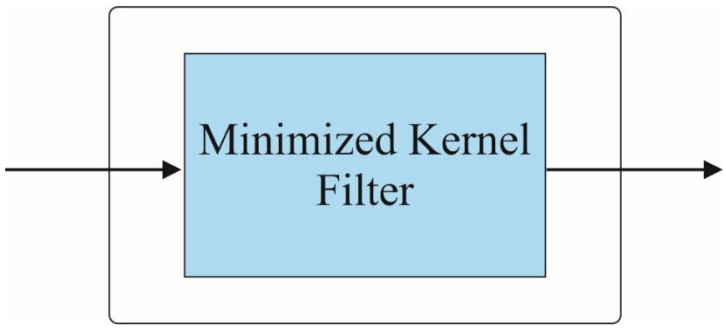


Fig 5.10: Minimized Hardware Kernel

5.4.6 Hardware Minimization Reports

1. Gray Image

The synthesis has been implemented on one plane (gray) image. The summery of utilization report after minimization is given in table 5.5.

Table 5.5 New Hardware Device Utilization Summary for Gray Image.

Logic Utilization	Used	Available	Utilization	Relative H/W Reduction
Number of Slice Registers	497	18224	2%	44%
Number of Slice LUTs	264	9112	2%	45%
Number of fully used LUT-FF pairs	188	573	32%	3%
Number of bonded IOBs	113	232	48%	15%
Number of BUFG/BUFGCTRLs	1	16	6%	0%
Number of DSP48A1s	7	32	21%	41%

2. RGB Image

The synthesis has been implemented on three planes (RGB) image. The summery of utilization report after minimization is given in table 5.6.

Table 5.6 Minimization Hardware Device Utilization Summary for Three planes (RGB) image.

Logic Utilization	Used	Available	Utilization	Relative H/W Reduction
Number of Slice Registers	1405	18224	7%	3%
Number of Slice LUTs	732	9112	8%	8%
Number of fully used LUT-FF pairs	516	1621	31%	6%
Number of bonded IOBs	217	232	93%	9%
Number of BUFG/BUFGCTRLs	1	16	6%	0%

3. YCbCr Image

On YCbCr color space the synthesis has been implemented on one plane (Y plane) only. The summery of utilization report after minimization is given in table 5.7.

Table 5.7 Minimization Hardware Device Utilization Summary for YCbCr

Logic Utilization	Used	Available	Utilization	Relative H/W Reduction
Number of Slice Registers	529	18224	2%	4%
Number of Slice LUTs	296	9112	3%	7%
Number of fully used LUT-FF pairs	220	605	36%	6%
Number of bonded IOBs	147	232	63%	0.67%
Number of BUFG/BUFGCTRLs	1	16	6%	0%
Number of DSP48A1s	7	32	21%	0%

5.4.7 Results

The hardware cosimulation system consists of PC and Xilinx Spartan-6 SP601 Evaluation Kit as shown in Figure 5.11. Two monitors have been used to demonstrate the cosimulation; the first is used to display Matlab XSG model, whereas the other is used to display the source video and the processed video on SP601 kit. The simulation waveforms using Xilinx ISE14.5 is illustrated in Figure 5.12. Samples of input and output video frames is shown in Table 5.8.

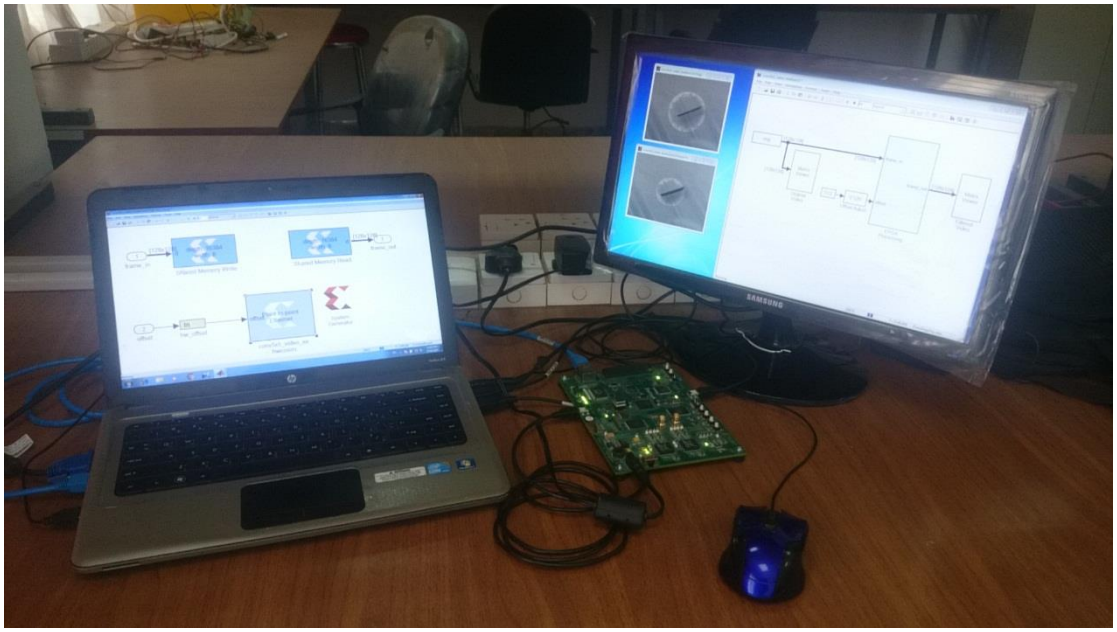


Fig 5.11: System Hardware

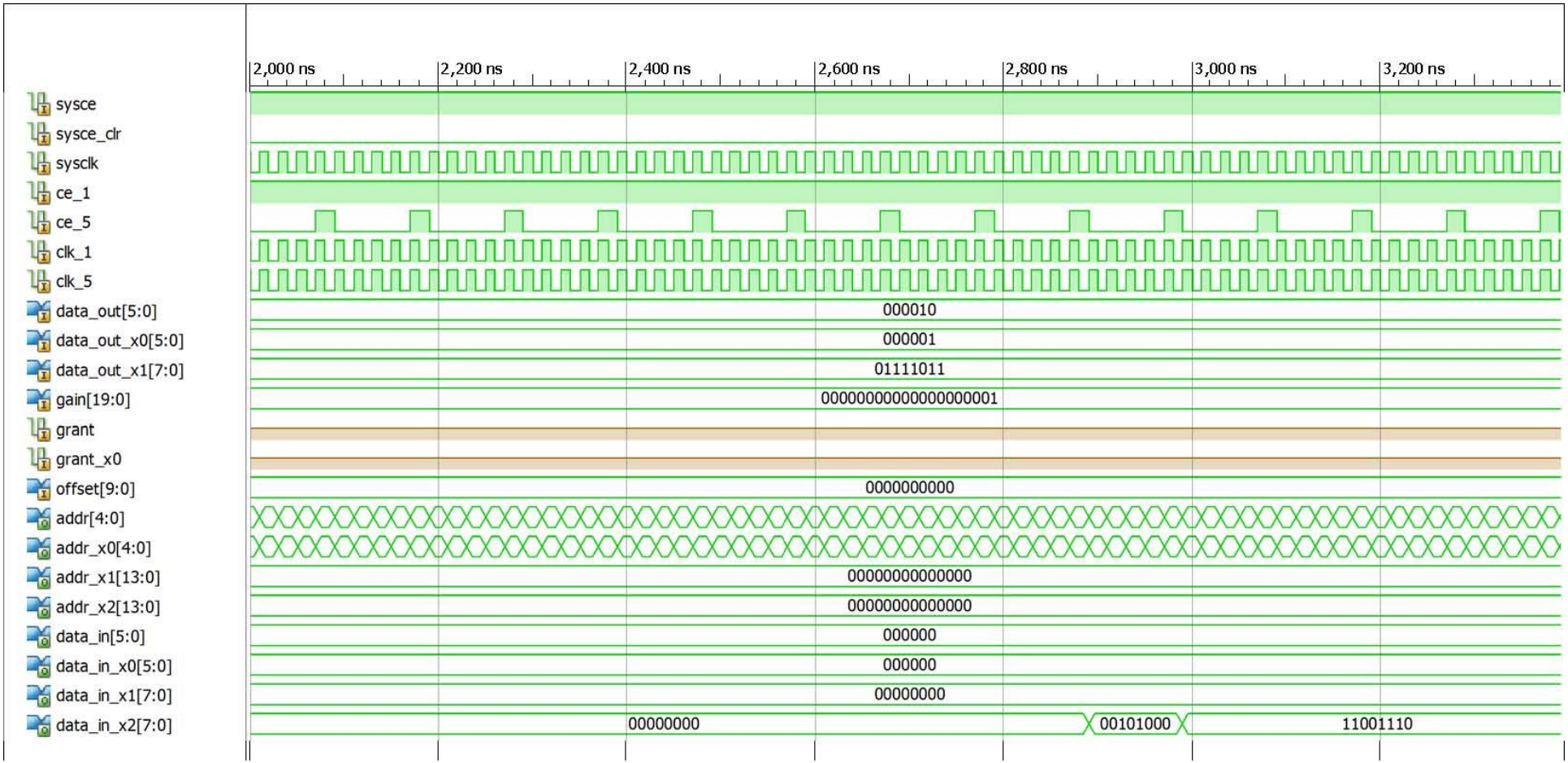
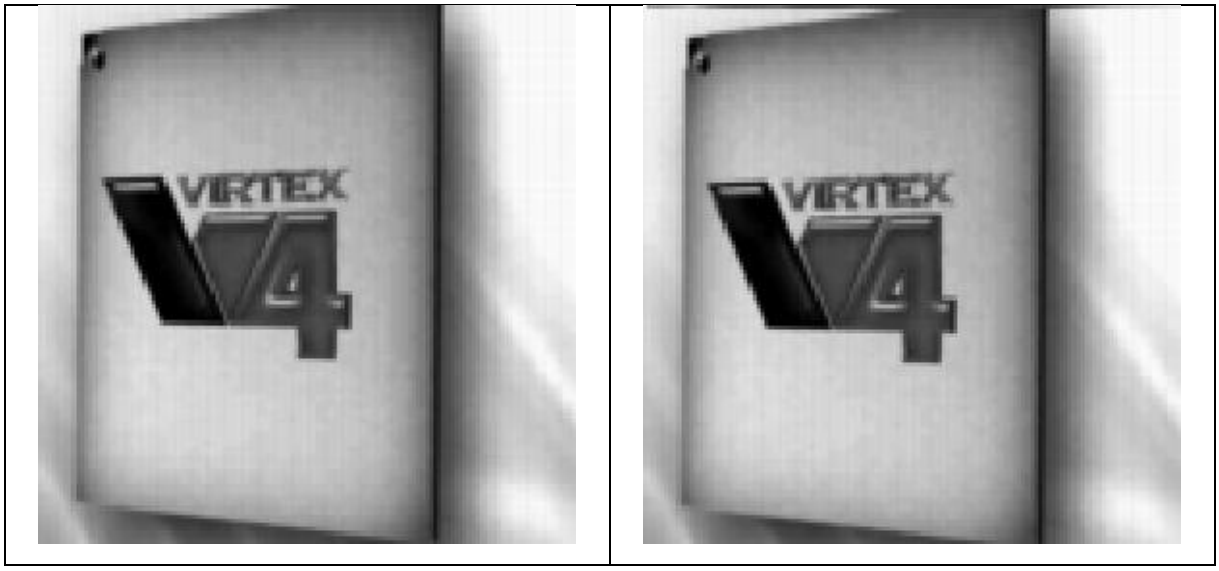


Fig 5.12: Hardware system waveforms

Table 5.8: XSG video frame samples**.**

Input Frame	Output Frame
	
	
	



5.5 Video Processing Acceleration Using FPGA-in-the-Loop

FPGA-in-the-Loop (FIL) accelerates a video processing simulation using Simulink and FPGA [15]. The FIL process consists of three steps. The first step, the design is simulated using Matlab Simulink. The FIL 2D-FOD deblurring Simulink model is illustrated in Figure 5.13. Then, cosimulation based ModelSim is generated, the cosimulation verification model is shown in Figure 5.14. The cosimulation model includes a cosimulation block which is EDA Simulink gateway to ModelSim that provides on the fly hardware signals. The hardware signals timing can be verified and investigated for timing and functional correctness. Figure 5.15 shows the timing signals using ModelSim. The last step, FIL verification, in which the hardware is implemented and simulated on the FPGA.

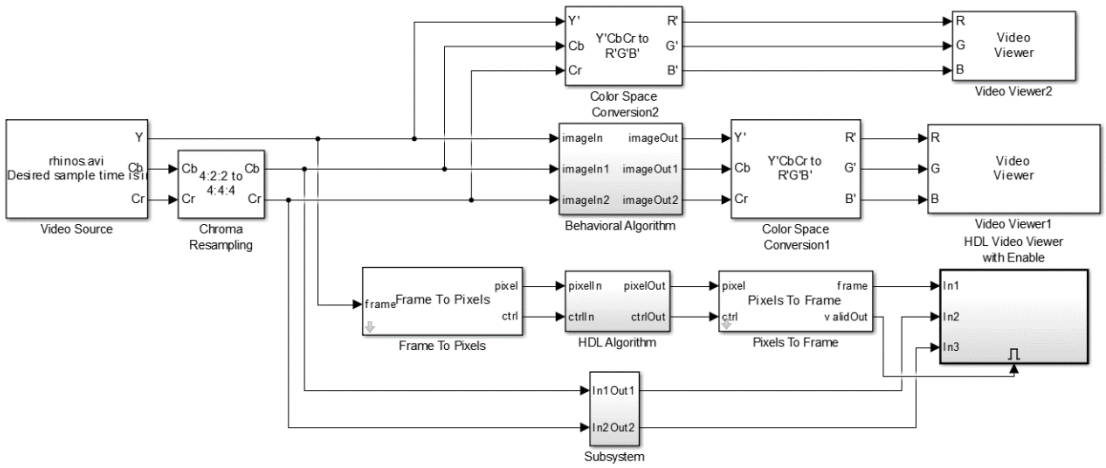


Fig 5.13: FIL 2D-FOD deblurring model.

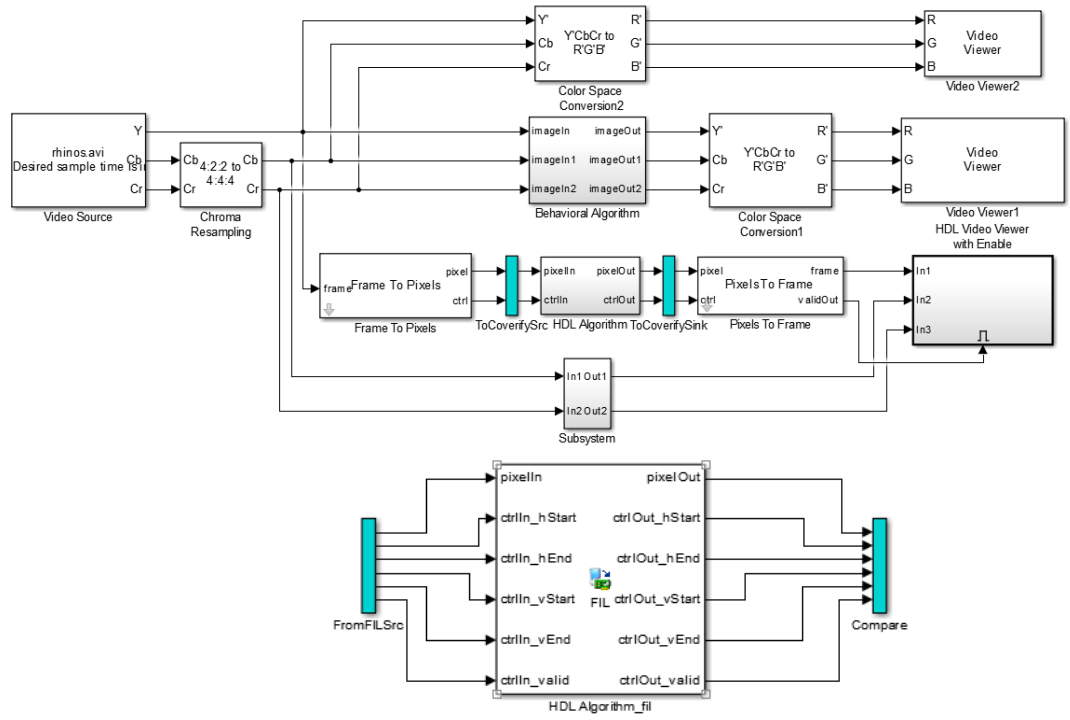


Fig 5.14: FIL 2D-FOD deblurring cosimulation model.

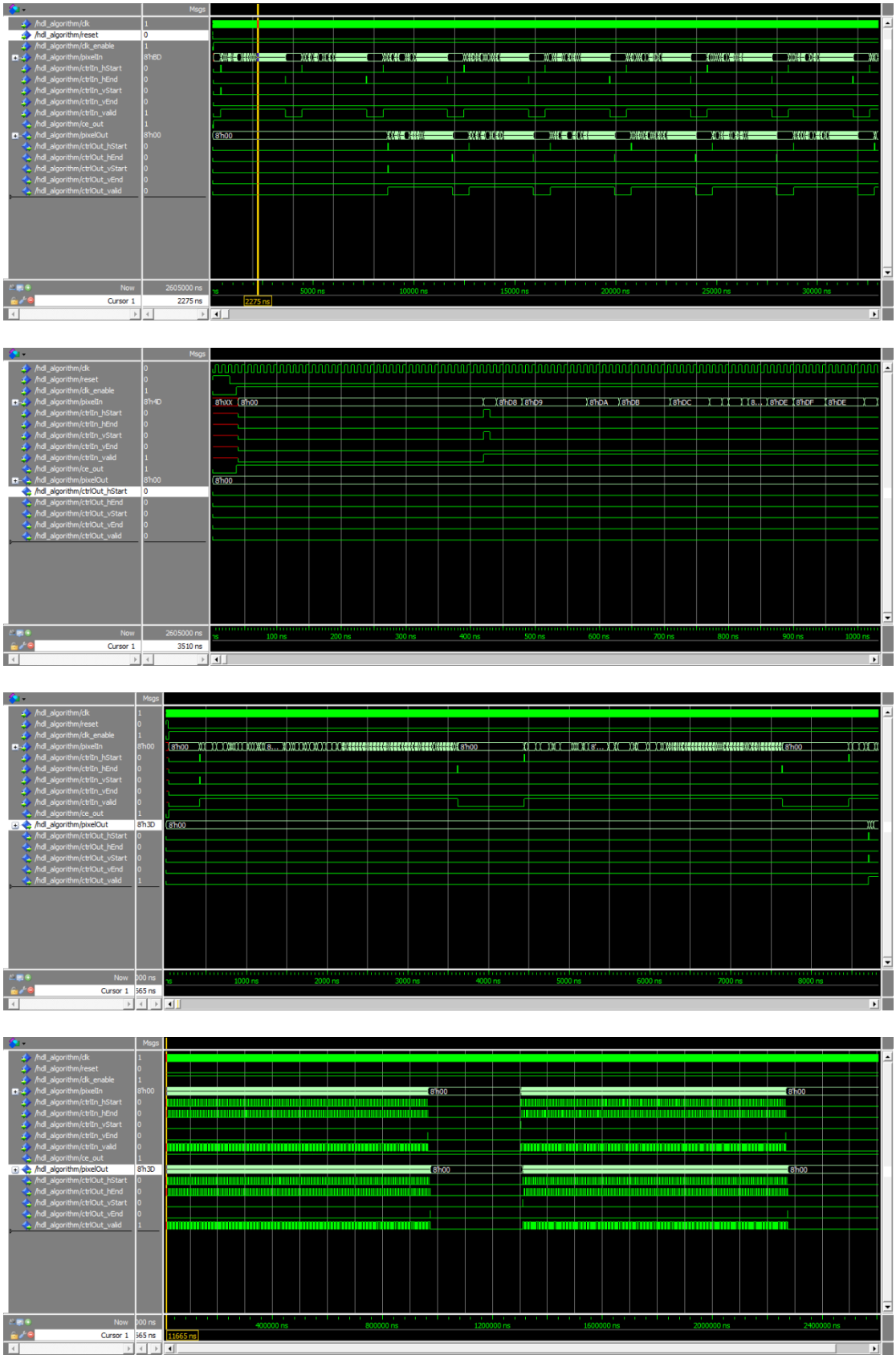


Fig 5.15: ModelSim timing signals.

5.5.1 FIL 2D-FOD deblurring Model

Matlab Vision HDL Toolbox provides a design environment that support video and image processing in HDL architecture as illustrated in Figure 5.13. The Video Source block provides frame stream to Frame to Pixel block. The Frame to Pixel block converts the incoming frame stream into stream of pixels and generates control signals for pixels stream (hStart, hEnd, vStart, vEnd, and valid data), also the size of the frame can be specified in this block. The HDL block incorporates the 2D-FOD deblurring Algorithm which processes the video in pixel by pixel. HDL block parameters includes line buffer size, padding method, fixed point arithmetic parameters, and 2D filter coefficients. The Pixel to Frame block convert the processed pixel stream into frame stream.

5.5.2 FIL Hardware Implementation Reports

1. Gray Frame

The synthesis has been implemented on one plane (gray) frame 480x640 frame size. The summery of utilization report after minimization is given in table 5.8.

Table 5.9 FIL Device Utilization Summary for Gray Frame.

Logic Utilization	Used	Available	Utilization
Number of Slice Registers	2622	18224	14%
Number of Slice LUTs	3882	9112	42%
Number of fully used LUT-FF pairs	1854	4650	39%
Number of bonded IOBs	30	232	12%
Number of BUFG/BUFGCTRLs	5	16	31%
Number of DSP48A1s	2	32	6%

Table 5.10 FIL Device Utilization Summary for Gray 1080x1920 Frame.

Logic Utilization	Used	Available	Utilization
Number of Slice Registers	2637	18224	14%
Number of Slice LUTs	3903	9112	42%
Number of fully used LUT-FF pairs	1869	4671	40%
Number of bonded IOBs	30	232	12%
Number of BUFG/BUFGCTRLs	5	16	31%
Number of DSP48A1s	0	32	0%

2. YCbCr Image

On YCbCr color space the synthesis has been implemented on one plane (Y plane) only. The summery of utilization report for 240x320 frame size is shown in table 5.9.

Table 5.11 FIL YCbCr Device Utilization Summary.

Logic Utilization	Used	Available	Utilization
Number of Slice Registers	3663	18224	20%
Number of Slice LUTs	4310	9112	47%
Number of fully used LUT-FF pairs	2169	5804	37%
Number of bonded IOBs	30	232	12%
Number of BUFG/BUFGCTRLs	5	16	31%
Number of DSP48A1s	2	32	6%

5.5.3 Results

The hardware cosimulation system consists of PC and Xilinx Spartan-6 SP601 Evaluation Kit as shown in Figure 5.x. Two monitors have been used to demonstrate the cosimulation; the first is used to display Matlab FIL model, whereas the other is used to display the source video and the processed video on SP601 kit. Samples of input and output video frames is shown in Table 5.11.

Table 5.12: Cat video frames K=0.5, v=0.6

Frame No.	Input Frame	Output Frame
1		
20		
40		

60		
90		
100		
120		
130		

160		
180		
190		
200		
210		

Chapter Six

Conclusions and Suggestions for Future Work

1.1 Conclusions

Video processing is a powerful branch of signal processing, which usually includes video filters. In current thesis a new deblurring algorithm based on 2D-FOD and modified Van Cittert algorithm has been proposed. The parameters of the proposed algorithm have been optimized using GA. Two approaches of the hardware implementation for the proposed algorithm based on Xilinx Spartan 6 (SP601) have been presented. From the design, analysis, and hardware implementation, one can conclude the following remarks that related with the different aspects of the subject:

1. The enhanced Van Cittert is an iterative algorithm which requires prior knowledge about the PSF, whereas the proposed is blind and non-iterative.
2. Yi-Fei commented that the Yi-Fei-2 mask operator is best filter mask for image enhancement applications but not suggested the value of the order of differentiation. The order of differentiation with the gain are optimized using genetic algorithm.
3. The cost function of the genetic algorithm is a fuzzy set of the contributions of entropy and PSNR, which gives better quality than using one of them alone.
4. The analyses of the proposed algorithm has been better to evaluate the results using the implemented GUI software application.

5. The online questionnaire survey is best than the traditional survey for visual quality evaluation. The traditional survey has many drawbacks for visual qualification. The Google forms is a creative platform for surveys. The people who contributed in questionnaire stated the recovered images is better than blurred version. 88.4% of the people state that the proposed algorithm is better than the conventional integer order operators.
6. Two approaches have been used for FPGA hardware implementation of the proposed algorithm using Xilinx system generator and FPGA in the loop. 44%, 45%, 15%, and 41% of hardware reduction in number of registers, LUTs, IOB, and DSP48A coprocessor have been achieved by reordering and simplifying the equations of the proposed model.
7. In Xilinx system generator methodology, a resolution 128x128 has been achieved on Spartan 6 (SP601) at 24 frame rate, whereas in FPGA in the loop a resolution of full high definition (1080x1920) attend on the same development kit at 30 frame rate.
8. ModelSim cosimulation verifier is better than Xilinx simulator. FIL with ModelSim provides on the fly hardware signals. The hardware signals timing can be verified and investigated for timing and functional correctness

1.2 Suggestions for Future Work

The following are some guidelines for future work:

1. Implementation of the proposed algorithm using system on chip (SoC) based FPGA, on Zynq-7000 evaluation kit.
2. Implementation of algorithm parameters estimation on FPGA.

REFERENCES

[1]. Guojun Lu, "*Advances in Digital Image Compression Techniques*", Computer Communications Vol. 16, No. 4, pp. 202-214, April 1993.

[2]. Wei Wang and Peizhong Lu, "*A New Image Deblurring Method Based on Fractional Differential*", International Conference on Audio, Language and Image Processing (ICALIP) IEEE Conferences, pp. 497 – 501, 2012.

[3]. Sitara K and Remya S, "*Image Deblurring in Bayesian Framework Using Template Based Blur Estimation*", International Journal of Multimedia and Its Applications (IJMA), Vol. 4, No. 1, February 2012.

[4]. Sabah F. Hamood, Mohd Shafry Mohd Rahim, Omar Farook and Daud Kassmuni, "*A Survey on Various Image Deblurring Methods*", Journal of Engineering and Applied Sciences, Vol. 11, No. 3, pp. 561-569, 2016.

[5]. I. M. El-Henawy, A. E. Amin, Kareem Ahmed, Hadeer Adel, "*A Comparative Study On Image Deblurring Techniq*", International Journal of Advances in Computer Science and Technology (IJACST), Vol. 3, No.12, pp. 01-08, 2014.

[6]. Per Christion Hansen, James G.Nagy, and Dianne P.O'Leary, "*Deblurring Image Matrices, Spectra, and Filtering*", Society for Industrial and Applied Mathematics (siam) Philadelphia, 2006.

[7]. Dhanabal R, Bharathi V, And S.Kartika, "*Digital Image Processing Using Sobel Edge Detection Algorithm in FPGA*", Journal of Theoretical and Applied Information

Technology (JATIT), Vol. 58, No. 1, pp. 130-139, December 2013.

[8]. Daggu V. Rao, Shruti Patil, Naveen Anne Babu and V Muthukumar, "*Implementation and Evaluation of Image Processing Algorithms on Reconfigurable Architecture using C-based Hardware Descriptive Languages*", International Journal of Theoretical and Applied Computer Sciences (ijtacs), Vol. 1 No 1, pp. 9–34, 2006.

[9]. Sio-Iong Ao, Len Gelman, "*Electronic Engineering and Computing Technology*", Springer, 2010.

[10]. A.C.Suthar, Mohammed Vayada, C.B.Patel, G.R.Kulkarni, "*Hardware Software co-simulation for Image Processing Applications*", IJCSI International Journal of Computer Science Issues, Vol. 9, No 2, pp. 560-562, March 2012.

[11]. T. Saidani, D. Dia, W. Elhamzi, M. Atri and R. Tourki, "*Hardware Co-simulation for Video Processing Using Xilinx System Generator*", Proceedings of the World Congress on Engineering, Vol. 1, 2009.

[12]. Uwe Meyer-Baese, "*Digital Signal processing with filed programable gate array*", Springer, 2001.

[13]. Jun Kong, Kesai Lu, and Min Jiang "*A New Blind Deblurring Method via Hyper-Laplacian Prior*", Procedia Computer Science Vol. 107 pp.789 – 795, 2017.

[14]. Hongyan Wang, Jinshan Pan, Zhixun Su, and Songxin Liang, " *Blind image deblurring using elastic-net based rank prior* ", Computer Vision and Image Understanding CVIU, Dec 2017.

[15]. Dong-Bok Lee, Shin-Cheol Jeong, Yun-Gu Lee, and Byung Cheol Song, "*Video Deblurring Algorithm Using Accurate Blur Kernel Estimation and Residual Deconvolution Based on*

a Blurred- unblurred Frame Pair", IEEE Transactions on Image Processing, Vol. 22, No. 3, pp 926-940, March 2013.

[16]. Mohammed Alareqi, R. Elgouri, and M. Tarhda, K. Mateur, A. "*Design and FPGA Implementation of Real-Time Hardware Co-Simulation for Image Enhancement in Biomedical Applications*", International Conference on Wireless Technologies, Embedded and Intelligent Systems (WITS), 2017.

[17]. A M Deshpande, and S Patnaik, "*A Qualitative-Quantitative Comparison of Image Motion Deblurring Algorithms*", International Conference and Workshop on Emerging Trends in Technology (ICWET), 2011.

[18]. Yi-Fei Pu, Ji-Liu Zhou, and Xiao Yuan, "*Fractional Differential Mask: A Fractional Differential-Based Approach for Multiscale Texture Enhancement*", IEEE Transactions On Image Processing, Vol. 19, No. 2, February 2010.

[19]. Wenqi Ren, Jinshan Pan, Xiaochun Cao, and Ming-Hsuan Yang, "*Video Deblurring via Semantic Segmentation and Pixel-Wise Non-Linear Kernel*", IEEE International Conference on Computer Vision (ICCV), pp. 1086 – 1094, 2017.

[20]. M. Lopez-Ramirez, L. M. Ledesma-Carrillo, E. Cabal-Yepez, G. Botella, C. Rodriguez-Donate, and Sergio Ledesma, "*FPGA-Based Methodology for Depth-of-Field Extension in a Single Image*", Digital Signal Processing, Vol.70, pp 14-23, Nov 2017.

[21]. Priyanka Raina, Mehul Tikekar, and Anantha P. Chandrakasan, "*An Energy-Scalable Accelerator for Blind*

Image Deblurring", IEEE Journal of Solid-State Circuits, Vol. 52, No. 7, pp. 1849 – 1862, July 2017 .

[22]. G. Bharadwaja Reddy, and K. Anusudha, "*Implementation of Image Edge Detection on FPGA using XSG*", International Conference on Circuit, Power and Computing Technologies [ICCPCT], 2016.

[23]. Chien-Cheng Tseng, and Shyi-Chyi Cheng, "*Digital Color Image Sharpening Using Fractional Differentiation and Discrete Cosine Transform*", International Symposium on Communications and Information Technologies (ISCIT) IEEE Conferences, pp. 181 – 186, 2012.

[24]. Chien-Cheng Tseng, and Su-Ling Lee, "*Color Image Sharpening Based on Fractional Differentiation and Modulation Transfer Function*", International Symposium on Consumer Electronics (ISCE) IEEE Conferences, pp. 201 – 202, 2013.

[25]. PU YiFei, Wang Weixing, Zhou Jiliu, Wang Yiyang, And Jia Huading, "*Fractional differential approach to detecting textural features of digital image and its fractional differential filter implementation*", Sci China Ser F-Inf Sci, Vol. 51, No. 9, pp. 1319-1339, Sep. 2008.

[26]. Thusitha Chandrapala , Amila Cabral, Thilina Sameera, Spumal Ahangama and Jayathu Samarawickrama, "*Hardware Implementation of Motion Blur removal*", 22nd International Conference on Field Programmable Logic and Applications (FPL) IEEE Conferences, pp. 243 – 248, 2012.

[27]. Zohair Al-Ameen, Ghazali Sulong and Md. Gapar Md. Johar, "*A Comprehensive Study on Fast image Deblurring*

Techniques", International Journal of Advanced Science and Technology Vol. 44, July, 2012.

[28]. Emrah ONAT, "*FPGA Implementation of Real Time Video Signal Processing Using Sobel, Robert, Prewitt and Laplacian Filters*", 25th Signal Processing and Communications Applications Conference (SIU) IEEE Conferences, pp. 1 – 4, 2017.

[29]. Qi Yang, Dali Chen, Tiebiao Zhao, and YangQuan Chen, "*Fractional Calculus in Image Processing: A Review*", Fractional Calculus and Applied Analysis, Vol. 19, No. 3, 2016.

[30]. Taresh Singh, and B. M. Singh, "*Comparative Analysis of Image Deblurring Techniques*", International Journal of Computer Applications, Vol. 153, No. 5, November 2016.

[31]. LU YUAN, "*Image Deblurring*", Ph.D Thesis, Hong Kong University of Science and Technology, 2009.

[32]. Su Bolan, "*Document Image Enhancement*", Ph.D Thesis, National University of Singapore, 2012.

[33]. João P. Oliveira, Mário A. T. Figueiredo, and José M. Bioucas-Dias, "*Parametric Blur Estimation for Blind Restoration of Natural Images: Linear Motion and Out-of-Focus*", IEEE Transactions on Image Processing, Vol. 23, No. 1, pp. 466-477, JAN 2014.

[34]. Xue-fen Wan, and Yi Yang, Xin Lin, "*Point Spread Function Estimation For Noisy Out-of-focus Blur Image Restoration*", International Conference on Software Engineering and Service Sciences, IEEE Conferences, pp. 344 – 347, 2010.

[35]. Sunghyun Cho, Yasuyuki Matsushita,and Seungyong Lee, "*Removing Non-Uniform Motion Blur from Images*",

International Conference on Computer Vision, IEEE Conferences, pp. 1 – 8, 2007;

[36]. Shengyang Dai and Ying Wu, "*Motion from Blur*", Conference on Computer Vision and Pattern Recognition, IEEE Conferences, pp. 1 – 8, 2008.

[37]. Jong Min Lee, Jeong Ho Lee, Ki Tae Park, and Young Shik Moon, "*Image deblurring based on the estimation of PSF parameters and the post-processing*", Optik - International Journal for Light and Electron Optics, Vol. 124, No 15, pp. 2224-2228, August 2013.

[38]. Jian-Feng Cai, Hui Ji, Chaoqiang Liu, and Zuowei Shen, "*Blind motion deblurring using multiple images*", Journal of Computational Physics, Vol. 228, No 14, pp. 5057-5071, August 2009.

[39]. Jiaya Jia, "*Single Image Motion Deblurring Using Transparency*", IEEE Conference on Computer Vision and Pattern Recognition, pp. 1-8, 2007.

[40]. Yang shen, and Lizhuang ma, "*Detecting and Removing the Motion Blurring from Video Clips*", I.J.Modern Education and Computer Science, pp. 17-23, 2010.

[41]. Ryan Wen Liu, and Tian Xu, "*A Robust Alternating Direction Method for Constrained Hybrid Variational Deblurring Model*", Cornell University Library, Computer Vision and Pattern Recognition, 2013.

[42]. Uwe Schmidt, Carsten Rother, Sebastian Nowozin, Jeremy Jancsary, and Stefan Roth, "*Discriminative Non-blind Deblurring*", IEEE Conference on Computer Vision and Pattern Recognition, pp. 604 – 611, 2013.

[43]. R. C. Puetter, T.R. Gosnell, and Amos Yahil, "*Digital Image Reconstruction: Deblurring and Denoising*", Annual Review of Astronomy and Astrophysics, Vol. 43, pp. 139-194, 2005.

[44]. Dr. Jassim M. , Dunia S. Tahir, and Dr.Fadhil A. Ali. "*Restoration of Noisy Blurred Image*", Basrah Journal for Engineering Science No. 2, pp. 90-101, 2010.

[45]. Yu-Hua Fan, Xiao-Ming Wei, and Shi-Yin Qin, "*Fast and robust deblurring method with multi-frame images based on PSF estimation and total variation optimization*", Optik - International Journal for Light and Electron Optics, Vol, 124, No. 16, pp 2285-2291, August 2013.

[46]. Hui Ji and Kang Wang, "*A two-stage approach to blind spatially-varying motion deblurring*", Computer Vision and Pattern Recognition (CVPR), IEEE Conference, 2012.

[47]. Sudha Tiwari, Naazish Rahim, Sandeep Sahu, and Nikita Sharma, "*Multiple Blur Images Restoration- A Parallel Computing Approach*", Fourth International Conference on Computational Intelligence and Communication Networks, pp 374 – 377, 2012.

[48]. LE NGOC THUY, "*Line-Field Based Adaptive Image Model for Blind Deblurring*", Ph.D Thesis, National University of Singapore, 2010.

[49]. Zhuzhong YANG, Fangnian LANG, Xiaohong YU,and Yu ZHANG, "*The Construction of Fractional Differential Gradient Operator*", Journal of Computational Information Systems Vol. 7, No. 12, pp. 4328-4342, 2011.

[50]. Shantanu Das, "*Functional Fractional Calculus for System Identification and Controls*", Springer-Verlag Berlin Heidelberg, 2008.

[51]. Keith B. Oldham, And Jerome Spanier, "*The Fractional Calculus*", Academic Press, INC, 1974.

[52]. Ivo Petráš, "*Fractional-Order Nonlinear Systems*", Higher Education Press, Beijing and Springer-Verlag Berlin Heidelberg, 2011.

[53]. Changpin Li and Fanhai Zeng et al., "*Numerical Methods for Fractional Calculus*", CRC Press Taylor & Francis Group, 2015.

[54]. George A. Anastassio, and Ioannis K. Argyros, "*Intelligent Numerical Methods II: Applications to Multivariate Fractional Calculus*", Springer International Publishing Switzerland, 2016.

[55]. Dumitru Baleanu, Kai Diethelm, Enrico Scalas, and Juan J. Trujillo, "*Fractional Calculus: Models and Numerical Methods"*, 2nd edition, New Jersey : World Scientific, 2016.

[56]. Blas M. Vinagre, Yang Quan Chen, and Ivo Petr, "*Two direct Tustin discretization methods for fractional-order differentiator/integrator*", Journal of the Franklin Institute, Vol. 340, No. 5, pp. 349-362, August 2003.

[57]. CH. Lubich, "*Discretized Fractional Calculus",* Society for Industrial and Applied Mathematics Vol. 17, No. 3, pp. 704-719, May 1986.

[58]. Mohamad Adnan Al-Alaoui, "*Discretization Methods of Fractional Parallel PID Controllers*", IEEE International Conference on Electronics, Circuits and Systems - (ICECS), pp 327 – 330, 2009.

[59]. Mitchell Melanie, "*An Introduction to Genetic Algorithms*", MIT Press, 1998.

[60]. Goldberg D.E., "*Genetic Algorithms in Search Optimization and Machine learning*", Adison Wesley, New York, 1989.

[61]. Mazin Z. Othman, Emad A. Al-Sabawi, "*Fractional Order System Identification Based on Genetic Algorithms*", Journal of Engineering Science and Technology, Vol. 8, No. 6, pp. 713 – 722, 2013.

[62]. Slami Saadi, Abderrezak Guessoum, Maamar Bettayeb, "*ABC optimized neural network model for image deblurring with its FPGA implementation*", Microprocessors and Microsystems, Vol. 37, No. 1, pp. 52-64, February 2013.

Appendix A

Source Code

GA main program

```
format long;
clear;
warning off;
clc;
global Y;

%================================================
% read image
%================================================
filename=strcat(pwd,'\images\gray\lena.jpg');
f =imread(filename);

%================================================
%  Simulate a Blur
%================================================
% Simulate a blurred image that you might get from camera motion.  Create a
% point-spread function, |PSF|, corresponding to the linear motion across
% 9 pixels (|LEN=9|), at an angle of 5 degrees (|THETA=5|). To simulate
% the blur, convolve the filter with the image using |imfilter|.


 LEN = 9;
 Sigma = 5;
 PSF = fspecial('gaussian', LEN, Sigma);

 g = conv2(double(f), double(PSF), 'same');
 g(find(g >255))=255;


LastGeneration = 100;
Nn             =  16  ;                % Number of bits G 0.8 ,v 0.8
                                       % [8 8]         = [G v]
Mn             =  1000 ;                % Number of cromosomes

Prob           = 0.4;
M_Prob         = 0.6 ;
%================================================================
X      = CreatCromosomesX(Nn,Mn);
X(1,:) = cromsoume(0.996, 0.828);
X(2,:) = cromsoume(0.996, 0.828);
X(3,:) = cromsoume(0.996, 0.828);
X(4,:) = cromsoume(0.996, 0.828);
X(5,:) = cromsoume(0.996, 0.828);

ZZ=[];
Z  =[];
h = waitbar(0,'Please wait...');
for kk=1:LastGeneration
```

```
    kk
    %====================================
    [Fitn] = Fitness( X ,f,g );
     error = max( Fitn )
     Z     = [Z;error];
    if error >200
        break;
    end

     %====================================
     X         = RouletteSel( X , Fitn );
     X         = crossoverfun(X,Prob,kk);

    for i = 1 :round(M_Prob*Mn),
         j   = floor(rand*(Nn-1))+1;
         X(i,:)=mutation(X(i,:),j);
    end

    %================ Printing FOP ==================
    %  [Fitn Index] = sort( Fitn);
    %  c            = Crm_Val(1,X(Index( size(X,1) ) ,:));
    %================================================
    waitbar(kk / LastGeneration)
end
    [p i] = max(Y(:,3));
       G  = Y(i,1)
       v  = Y(i,2)

      mask =DFD2(5,v);


     dg = conv2(g,mask,'same');
     %rX = g - G*dg;
      rX=g + G*(g-dg);
     P1=psnr(double(f),g,255) ;
    subplot(1,3,1),imshow( f ); title(sprintf('source image'));
    subplot(1,3,2),imshow(uint8( g) ); title(sprintf('blured image=%g',P1));
    subplot(1,3,3),imshow( uint8(rX) ); title(sprintf('enhanced
image=%g',p));
```

Fitness Function

```
function [ b ] =costfunMix( x )
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here
global X;
global bX;
global Y;

G=x(1);
v=x(2);
    mask =DFD2(5,v);
    dg = conv2(bX,mask,'same');
    rX = bX + G*(bX-dg);
    P1=ntrop(rX,255) ;
    b1 = 1/P1;
```

```
    P2=psnr(double(X),rX,255) ;
    b2 = 1/P2;
     b = 99999;
     if P1 > 7
        if P2 > 75
            b = 1/(0.3*P1) + 1/(0.7*P2) ;
        end
     end

    Y = [Y ; G v P1 P2];

% rX= imfilter( bX , mask );

end
```

```
function [ mask ] = DFD2( N , v )
% The discrete form of fractional differentiation (DFD ) of order a
% presented byGrünwald "YiFeiPU-2"
% where:
% N : size of mask
% v : order
% Reference : Fractional Differential Mask: A fractional Differential-Based
% Aproch for Multiscale Texture Enhancement.

n = N-2;

w(1) = ( v/4 ) + ( v^2/8 ) ;
w(2) = 1 - ( v^2/2 ) - ( v^3/8 ) ;
w(3) = -(5*v/4) + (5*v^3/16) + (v^4/16) ;

for k = 2 : n-2
    w(k+2) = (1/gamma(-v))*((gamma(k-v+1)/factorial(k+1))* w(1) + ...
        ((gamma(k-v)/factorial(k))*(1-(v^2/4)))+...
        ((gamma(k-v-1)/factorial(k-1))*((-v/4)+(v^2/8))));
end
if N > 3
    w(N-1) = (1/gamma(-v))*((gamma(n-v-1)/factorial(n-1))*(1-(v^2/4)) + ...
              ((gamma(n-v-2)/factorial(n-2))*((-v/4)+(v^2/8))));
    w(N) = (1/gamma(-v))*((gamma(n-v-1)/factorial(n-1))*((-v/4)+(v^2/8)));
end

 L = zeros (N ,N) ;
 L( (N+1)/2 , : ) = w ;
 R = fliplr(L) ;
 U = flipud(L)' ;
 D = flipud(U) ;
 RU = diag(w) ;
 RD = flipud(RU) ;
 LU = fliplr(RU) ;
 LD = flipud(LU) ;

 mask = L + R + U + D + RU + RD + LU + LD ;
 mask = mask ./sum(sum(mask)) ;
end

function [ out ] = AD( img1 , img2 )
%Compute Average Difference (AD) between images
%img1 : Original Image
%img2 : Restoration Imag
%Ref. ABC optimized neural network model for image deblurring with its FPGA
implientation.
 [ M , N] = size( img1 ) ;
 img1 = double( img1 ) ;
 img2 = double( img2 ) ;
 out = sum( sum( ( img1 - img2 ) ./ ( M * N ) ) ) ;
end

function [ out ] = Img_MSE( img1 , img2 )
%Compute Mean Square Error ( MSE ) between images
%img1 : Original Image
%img2 : Restoration Imag
```

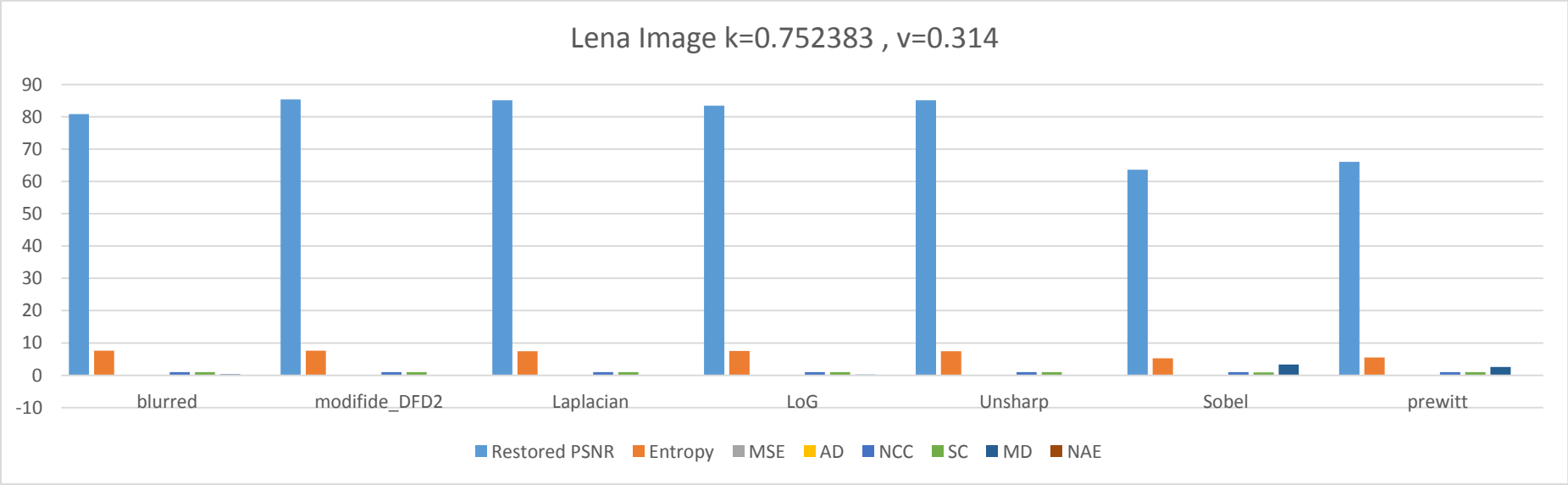
```
%Ref. ABC optimized neural network model for image deblurring with its FPGA
implientation.
 [ M , N] = size( img1 ) ;
 img1 = double( img1 ) ;
 img2 = double( img2 ) ;
 out = sum( sum( ( img1 - img2 ).^2 ) ) / ( M * N ) ;
end

function [ out ] = MD( img1 , img2 )
%Compute Maximume Difference ( MD ) between images
%img1 : Original Image
%img2 : Restoration Imag
%Ref. ABC optimized neural network model for image deblurring with its FPGA
implientation.
 [ M , N] = size( img1 ) ;
 img1 = double( img1 ) ;
 img2 = double( img2 ) ;
 out = max(max(abs( img1 - img2 )));
end


function [ out ] = NAE( img1 , img2 )
%Compute Normalized Absolute Error (NAE) between images
%img1 : Original Image
%img2 : Restoration Imag
%Ref. ABC optimized neural network model for image deblurring with its FPGA
implientation.
 [ M , N] = size( img1 ) ;
 img1 = double( img1 ) ;
 img2 = double( img2 ) ;
 out = ( sum( sum( abs( img1 - img2 ))) ) / (sum( sum( abs( img2 ) ) )) ;
end


function [ out ] = NCC( img1 , img2 )
%Compute Normalized Cross-Corelation ( NCC ) between images
%img1 : Original Image
%img2 : Restoration Imag
%Ref. ABC optimized neural network model for image deblurring with its FPGA
implientation.
 [ M , N] = size( img1 ) ;
 img1 = double( img1 ) ;
 img2 = double( img2 ) ;
 out = (sum( sum( ( img1 .* img2 ) ))) / ( sum(sum(img1.^2)) ) ;
end


function  h  =  ntrop (x,  n)
%NTROP  Computes  a  first - order  estimate  of  the  ent ropy  of  a  mat
rix .
%  H  =  NTROP (X,  N)  returns  the  ent ropy  of  matrix  X  with  N
%  symbols . N  =  256  if  omitted  but  it  must  be  larger  than  the
%  number  of  unique  values  in X  for  accu rate  results .  The  estimate
%  assumes  a  statistically  independent  source  characterized  by  the
%  relative  frequency  of  occu rrence  of  the  elements  in  X.
%  The  estimate  is a  lower  bound  on  the  average  number  of  bits  per
%  unique  value  (or  symbol )  when  coding  without  coding  redundancy .
```

```
error( nargchk (1 ,  2,  nargin) );  %  Check  input  arguments
if  nargin  < 2
n  =  256 ;  %  Default  for  n.
end
x  =  double(x) ;
xh =  hist(x( :),  n) ;
xh = xh  /  sum(xh( :));
%  Make  input  double
%  Compute  N - bin  histogram
%  Compute  probabilities
%  Make  mask  to  eliminate  O's  since  log2 (0)  = -inf .
i  =  find(xh) ;
h =  -sum (xh(i)  .*  log2 (xh(i )));  %  Compute  entropy


function [ PSNR_out ] = PSNR( img1 , img2 )
%Compute peak signal-to-noise ratio ( PSNR in db) between images
%img1 : Original Image
%img2 : Restoration Imag
 [ M , N] = size( img1 ) ;
 img1 = double( img1 ) ;
 img2 = double( img2 ) ;
 MSE = sum( sum( ( img1 - img2 ).^2 ) ) / ( M * N ) ;
%  R = max( max ( img1 ) ) ;
 PSNR_out = 10 * log10 ( 255^2 / MSE ) ;

end


function [ out ] = SC( img1 , img2 )
%Compute Structural Content ( SC ) between images
%img1 : Original Image
%img2 : Restoration Imag
%Ref. ABC optimized neural network model for image deblurring with its FPGA
implientation.
 [ M , N] = size( img1 ) ;
 img1 = double( img1 ) ;
 img2 = double( img2 ) ;
 out = (sum( sum( ( img1.^2 ) ) )) / (sum( sum( ( img2.^2 ) ) )) ;
end
```

Appendix B

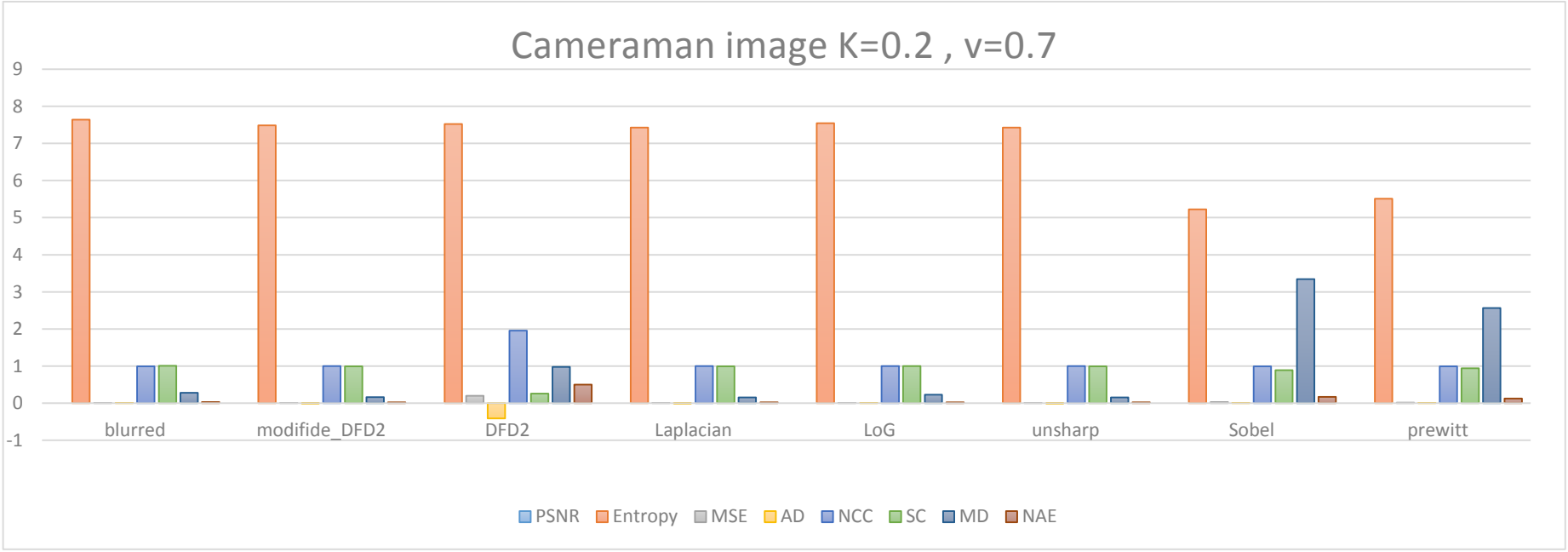
Lena Image k=0.752383 , v=0.314

Filter	Restored PSNR	Entropy	MSE	AD	NCC	SC	MD	NAE
blurred	80.790612	7.637887	0.000542	0.001128	0.992740	1.012031	0.284636	0.031078
modifide_DFD2	85.360605	7.574223	0.000189	-0.000028	0.998817	1.001449	0.178733	0.021010
Laplacian	85.116070	7.429287	0.000200	-0.001039	1.003726	0.991646	0.153924	0.020823
LoG	83.416621	7.546372	0.000296	0.000139	0.998792	1.000980	0.232835	0.024322
Unsharp	85.116070	7.429287	0.000200	-0.001039	1.003726	0.991646	0.153924	0.020823
Sobel	63.587622	5.222922	0.028465	0.000742	0.992300	0.890635	3.342715	0.168505
prewitt	66.084647	5.506724	0.016018	0.000839	0.992410	0.941101	2.567970	0.128824



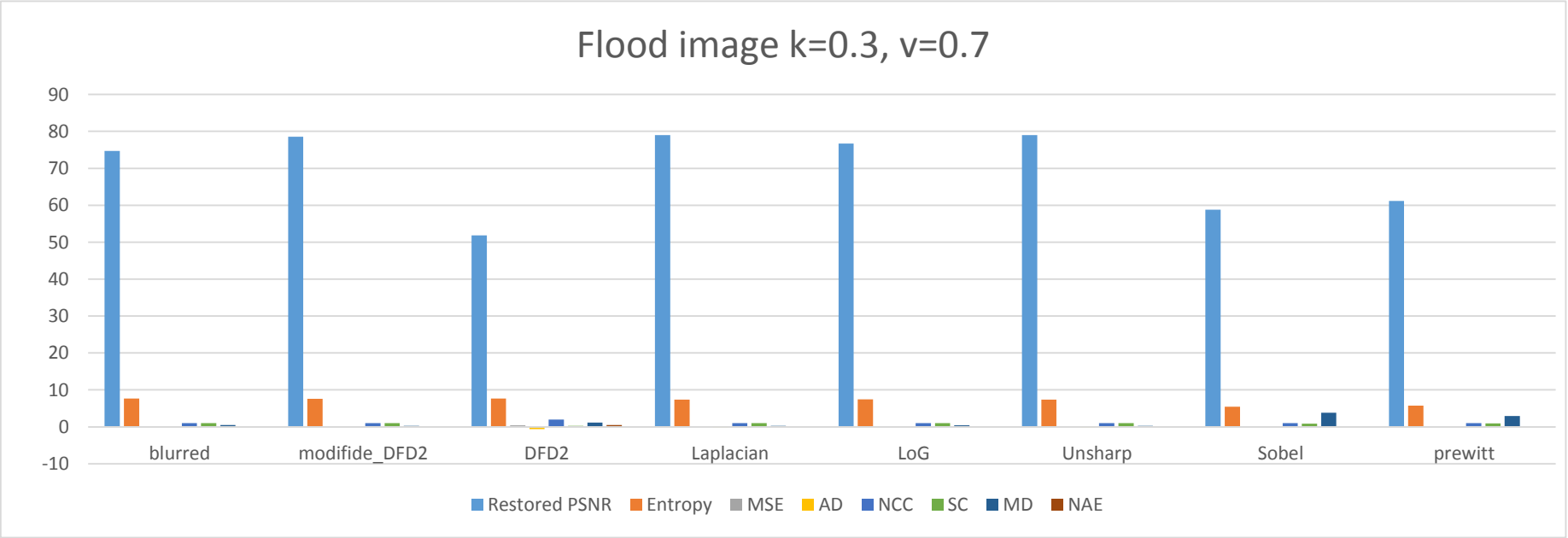
Cameraman image K=0.2 , v=0.7

Filter	Restored PSNR	Entropy	MSE	AD	NCC	SC	MD	NAE
blurred	79.961626	7.071894	0.000656	0.001538	0.991878	1.014048	0.333785	0.024468
modifide_DFD2	86.724179	6.934150	0.000138	-0.000304	1.001042	0.997418	0.163777	0.012385
DFD2	53.910048	7.078748	0.264287	-0.453616	1.947099	0.259242	1.329982	0.498752
Laplacian	85.042641	6.785014	0.000204	-0.001418	1.005435	0.988521	0.163123	0.013854
LoG	83.925065	6.890248	0.000263	0.000189	0.999456	1.000128	0.254538	0.014748
Unsharp	85.042641	6.785014	0.000204	-0.001418	1.005435	0.988521	0.163123	0.013854
Sobel	60.958385	4.946218	0.052149	0.000228	0.989803	0.854947	3.275105	0.189710
prewitt	63.433008	5.193887	0.029497	0.000555	0.990321	0.918995	2.360802	0.145477



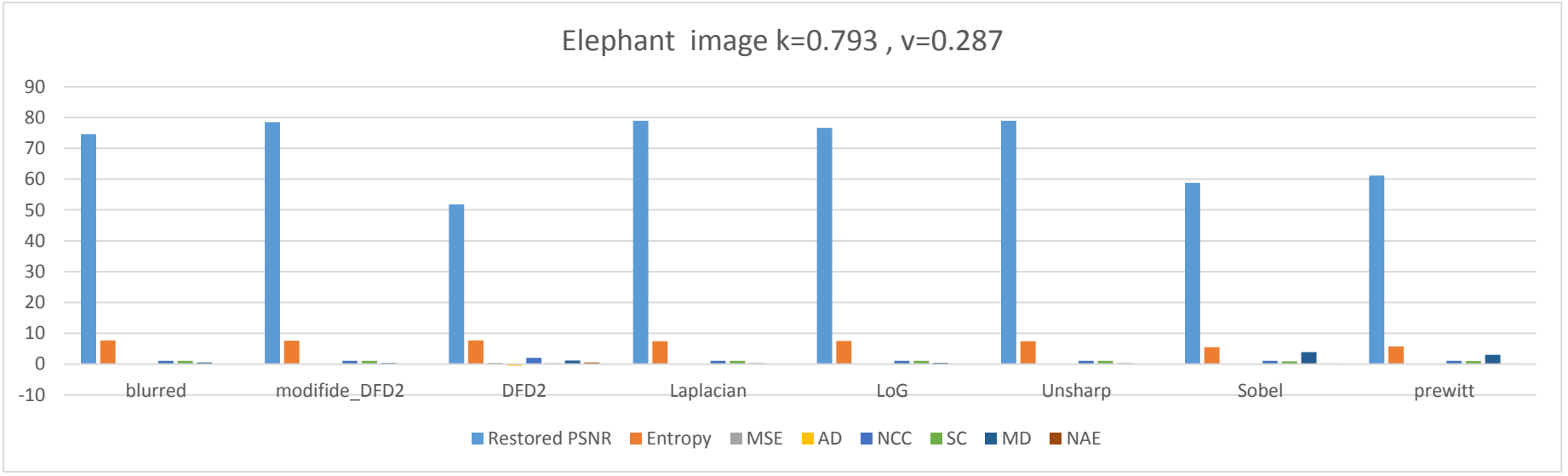
Flood image k=0.3, v=0.7

Filter	Restored PSNR	Entropy	MSE	AD	NCC	SC	MD	NAE
blurred	67.401483	2.859984	0.011829	0.002612	0.976721	1.031494	0.840845	0.042121
modifide_DFD2	68.878660	3.124310	0.008418	-0.001096	0.990544	1.007564	0.808030	0.038102
DFD2	49.580798	3.122114	0.716144	-0.795604	1.907365	0.264197	2.189519	0.498572
Laplacian	69.446296	2.816521	0.007387	-0.002405	0.994606	1.000780	0.808350	0.036722
LoG	68.223840	2.983758	0.009788	0.000323	0.985797	1.015377	0.840418	0.039971
Unsharp	69.446296	2.816521	0.007387	-0.002405	0.994606	1.000780	0.808350	0.036722
Sobel	56.365158	2.610431	0.150164	0.002612	0.976720	0.864379	3.655615	0.133830
prewitt	58.658782	2.686860	0.088553	0.002612	0.976720	0.931600	2.808835	0.107359



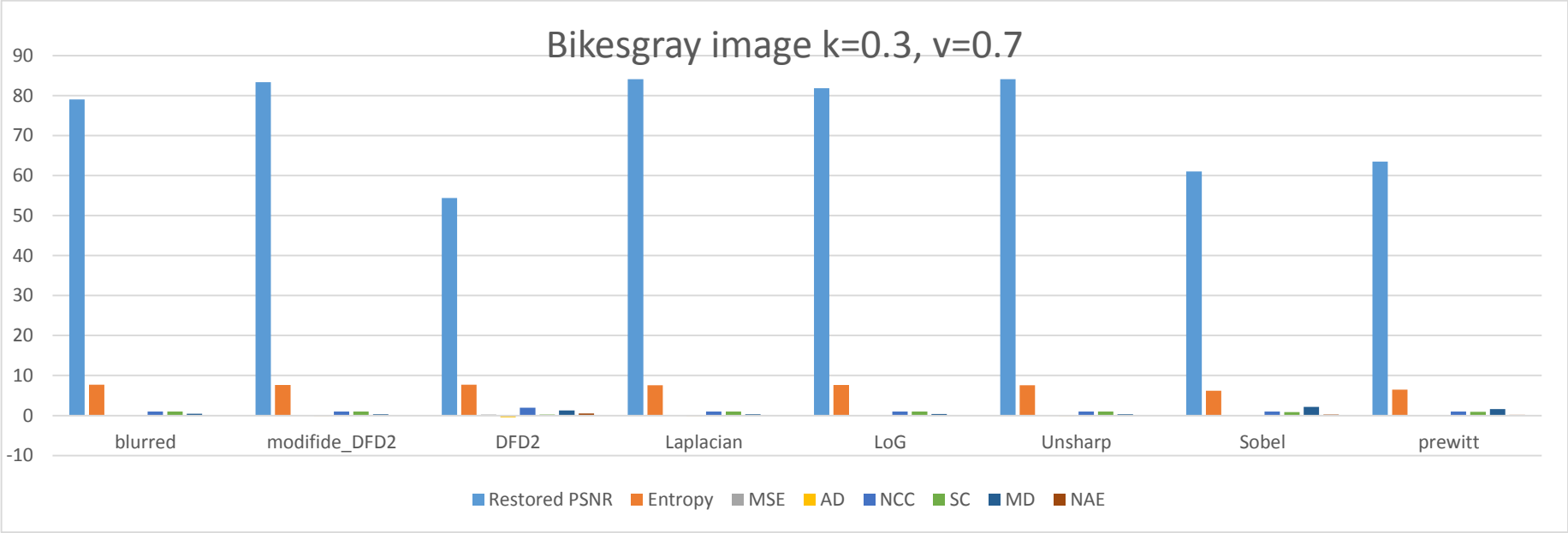
Elephant image k=0.793 , v=0.287

Filter	Restored PSNR	Entropy	MSE	AD	NCC	SC	MD	NAE
blurred	74.665128	7.632426	0.002221	0.002945	0.988582	1.018107	0.482551	0.034989
modifide_DFD2	78.570698	7.598604	0.000904	- 0.000027	0.997057	1.003845	0.293511	0.022771
DFD2	51.845102	7.649161	0.425177	- 0.603162	1.966471	0.256407	1.151010	0.496051
Laplacian	78.989112	7.370486	0.000821	-0.002709	1.004606	0.989042	0.266102	0.021352
LoG	76.695240	7.477546	0.001392	0.000366	0.996359	1.004134	0.395282	0.027626
Unsharp	78.989112	7.370486	0.000821	-0.002709	1.004606	0.989042	0.266102	0.021352
Sobel	58.784261	5.436831	0.086031	0.000519	0.985042	0.857802	3.853230	0.171632
prewitt	61.192517	5.750737	0.049412	0.001126	0.985928	0.922299	2.961191	0.132922



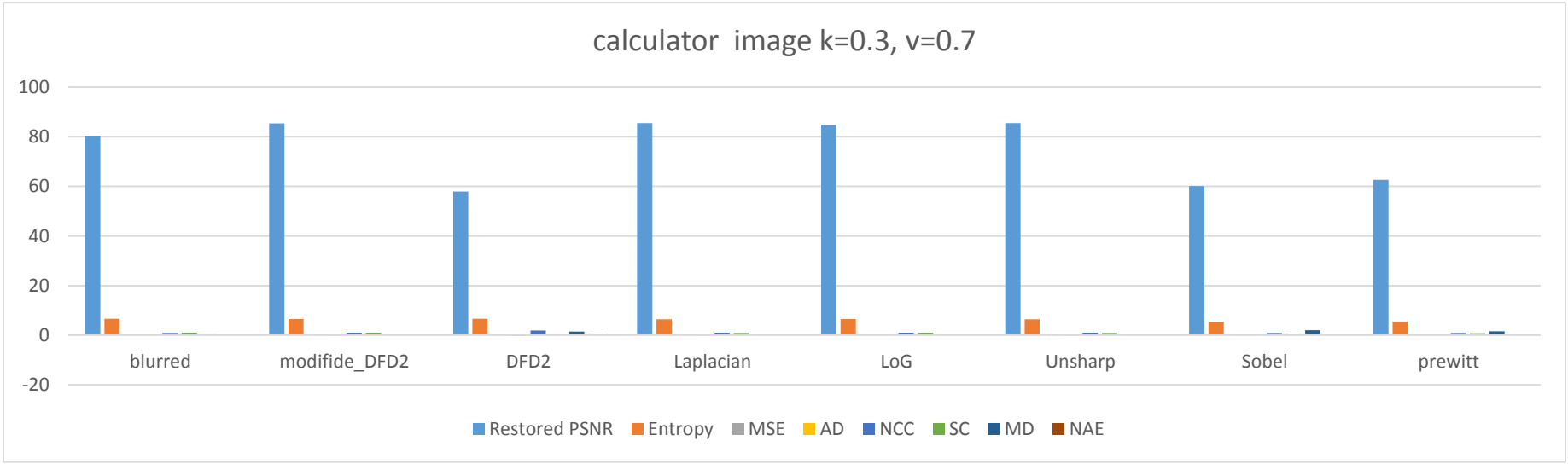
Bikesgray image k=0.3, v=0.7

Filter	Restored PSNR	Entropy	MSE	AD	NCC	SC	MD	NAE
blurred	79.043069	7.726895	0.000811	0.000806	0.992200	1.012416	0.450174	0.039206
modifide_DFD2	83.333983	7.645812	0.000302	-0.000360	1.000772	0.997222	0.300091	0.025879
DFD2	54.374432	7.670179	0.237486	-0.434671	1.955831	0.257111	1.287592	0.500073
Laplacian	84.085234	7.546028	0.000254	-0.000746	1.002600	0.993794	0.284949	0.023490
LoG	81.850115	7.616162	0.000425	0.000097	0.998198	1.001859	0.358445	0.029490
Unsharp	84.085234	7.546028	0.000254	-0.000746	1.002600	0.993794	0.284949	0.023490
Sobel	61.015093	6.203829	0.051472	0.000200	0.991605	0.836739	2.130504	0.260134
prewitt	63.490153	6.462877	0.029112	0.000352	0.991753	0.906325	1.635491	0.200417



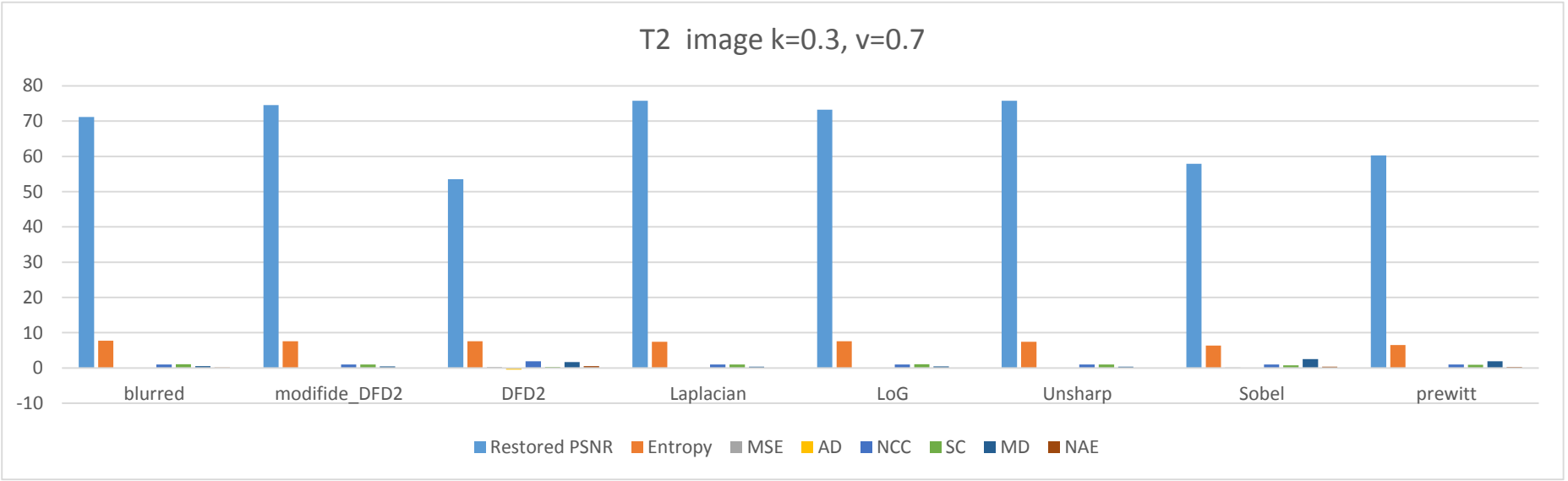
calculator image k=0.3, v=0.7

Filter	Restored PSNR	Entropy	MSE	AD	NCC	SC	MD	NAE
blurred	80.338649	6.672735	0.000601	0.000280	0.983257	1.029066	0.428060	0.039812
modifide_DFD2	85.383511	6.493748	0.000188	-0.000119	1.005509	0.987501	0.239548	0.024259
DFD2	57.867128	6.609769	0.106259	-0.292772	1.892343	0.269503	1.467084	0.499514
Laplacian	85.460809	6.398776	0.000185	-0.000259	1.008502	0.981724	0.215157	0.022976
LoG	84.693804	6.554018	0.000221	0.000034	0.999720	0.998640	0.313021	0.026501
Unsharp	85.460809	6.398776	0.000185	-0.000259	1.008502	0.981724	0.215157	0.022976
Sobel	60.069220	5.403374	0.063997	0.000210	0.983139	0.656215	2.039721	0.321344
prewitt	62.571629	5.603583	0.035968	0.000228	0.983168	0.781414	1.555946	0.252077



T2 image k=0.3, v=0.7

Filter	Restored PSNR	Entropy	MSE	AD	NCC	SC	MD	NAE
blurred	71.151117	7.696560	0.004988	0.000691	0.971412	1.041811	0.513711	0.096139
modifide_DFD2	74.524315	7.575623	0.002294	-0.000438	0.993614	1.004958	0.411985	0.066555
DFD2	53.520786	7.595339	0.289069	-0.475598	1.868819	0.268439	1.680386	0.500398
Laplacian	75.777750	7.439962	0.001719	-0.000780	0.999193	0.995758	0.365977	0.058078
LoG	73.235956	7.558643	0.003087	-0.000034	0.986495	1.016740	0.456417	0.076124
Unsharp	75.777750	7.439962	0.001719	-0.000780	0.999193	0.995758	0.365977	0.058078
Sobel	57.905682	6.312300	0.105320	-0.000540	0.970202	0.769094	2.490982	0.355072
prewitt	60.287675	6.493188	0.060857	-0.000232	0.970505	0.870376	1.887599	0.284722

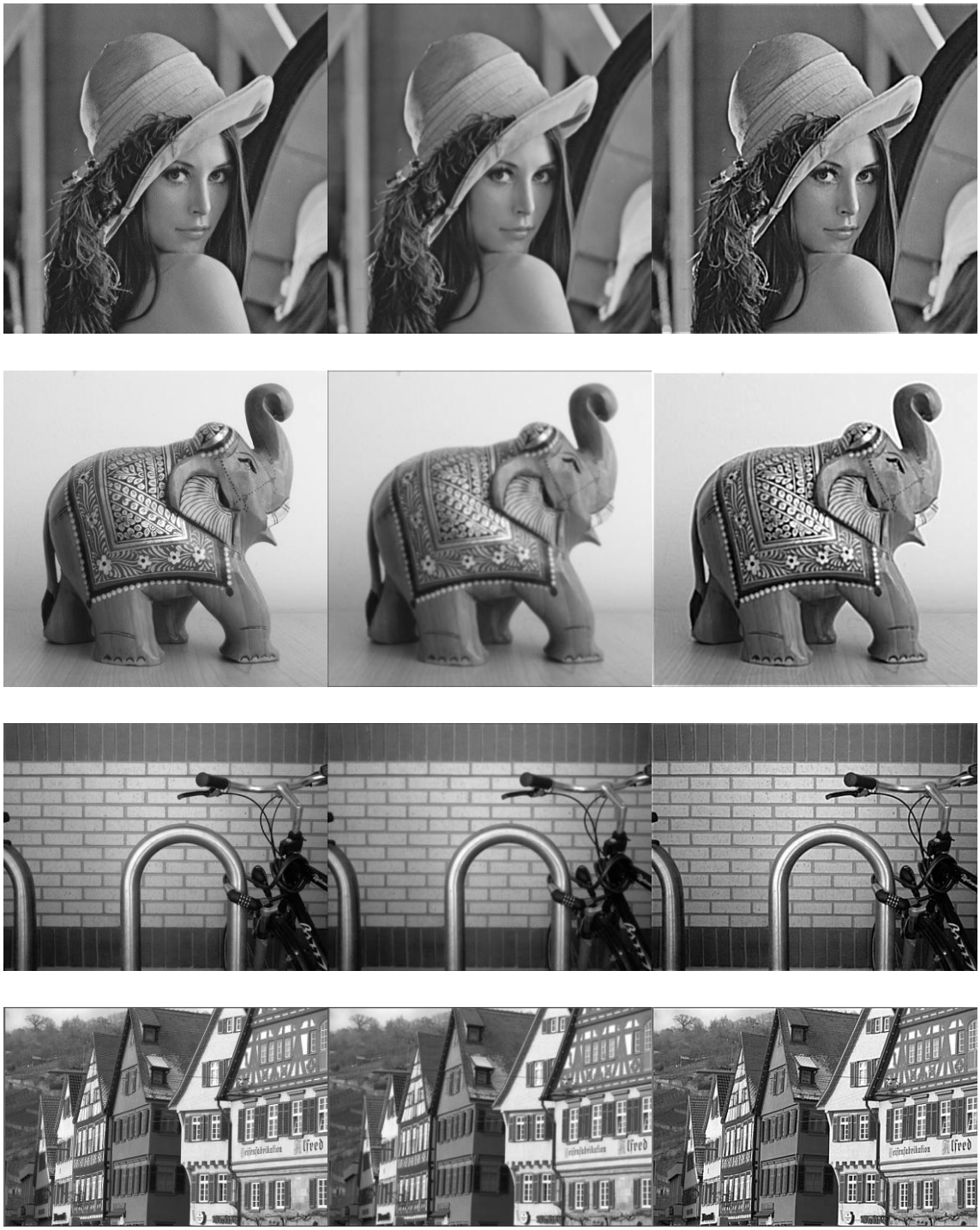


Appendix C

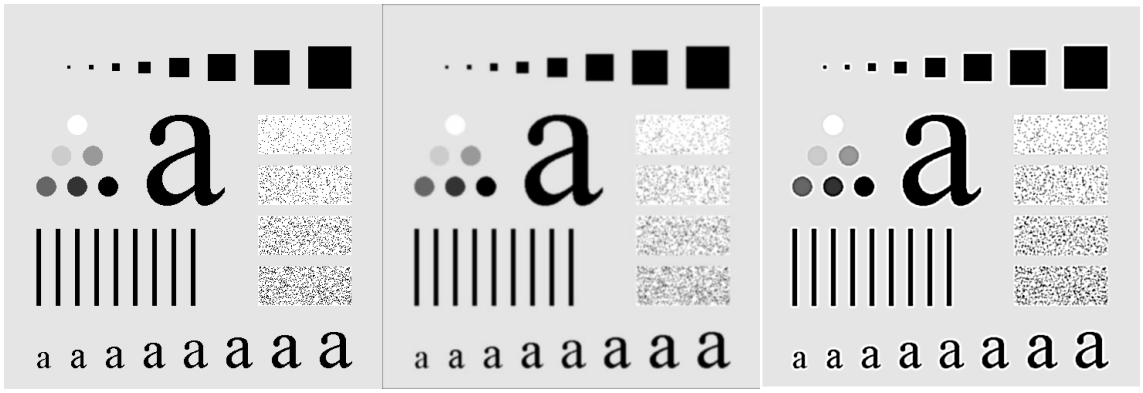
Gaussian blur using sigma $\sigma = 1$ and $11x11$ kernel size

Deblurring using modified DFD2 K=0.5 and v=0.75

Original image Blurred image Restored image




Σ+ 1/x √x LOG LN XEQ
COMPLEX STO RCL R↓ SIN COS TAN
ALPHA ENTER LAST x x≷y MODES +/- DISP E CLEAR
BST SOLVER ∫f(x) MATRIX STAT
7 8 9 ÷
SST BASE CONVERT FLAGS PROB
4 5 6 ×


a
a a a a a a a a

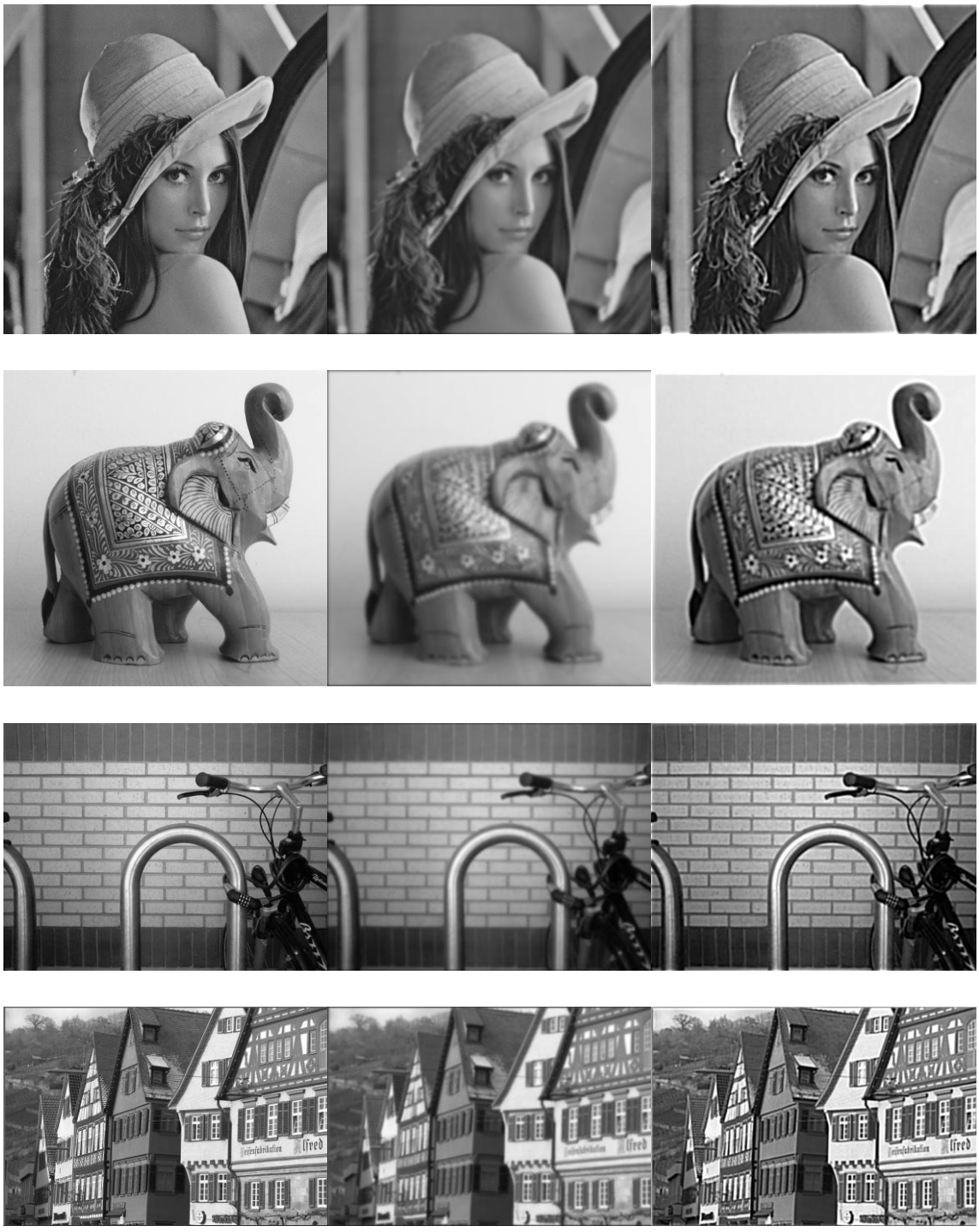
C2

C3

Gaussian blur using sigma $\sigma = 2$ and $11x11$ kernel size

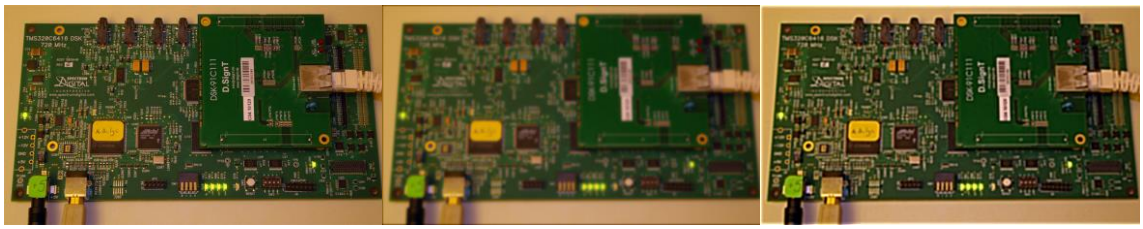
Deblurring using modified DFD2 K=0.8 and v=0.88

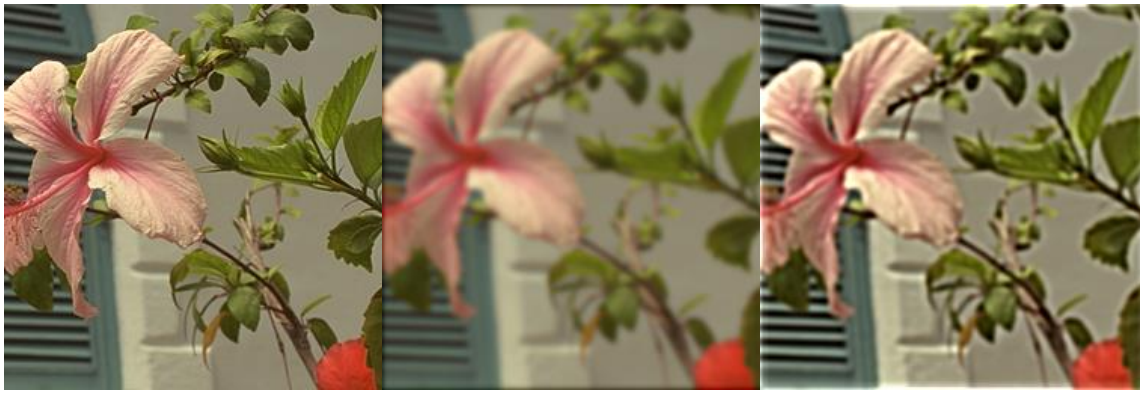
Original image Blurred image Restored image



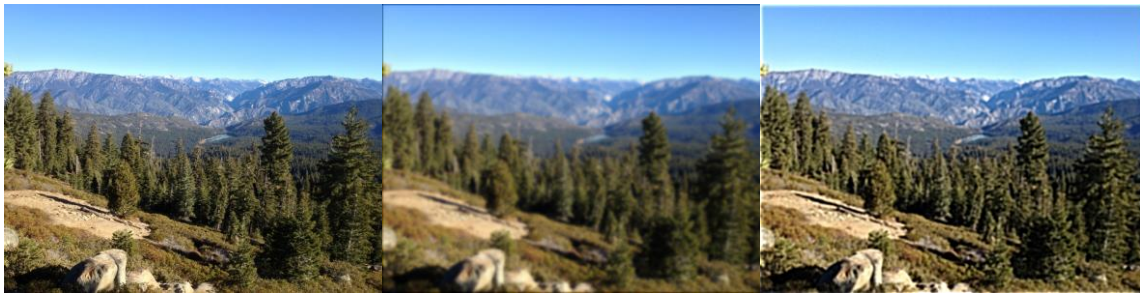

Σ+ 1/x √x LOG LN XEQ
STO RCL R↓ SIN COS TAN
ENTER x≷y +/- E
7 8 9 ÷
4 5 6 ×


a
a a a a a a a a






Bahamas
Bahamas
Bahamas



Gaussian blur using sigma $\sigma = 0.1$ and $11x11$ kernel size

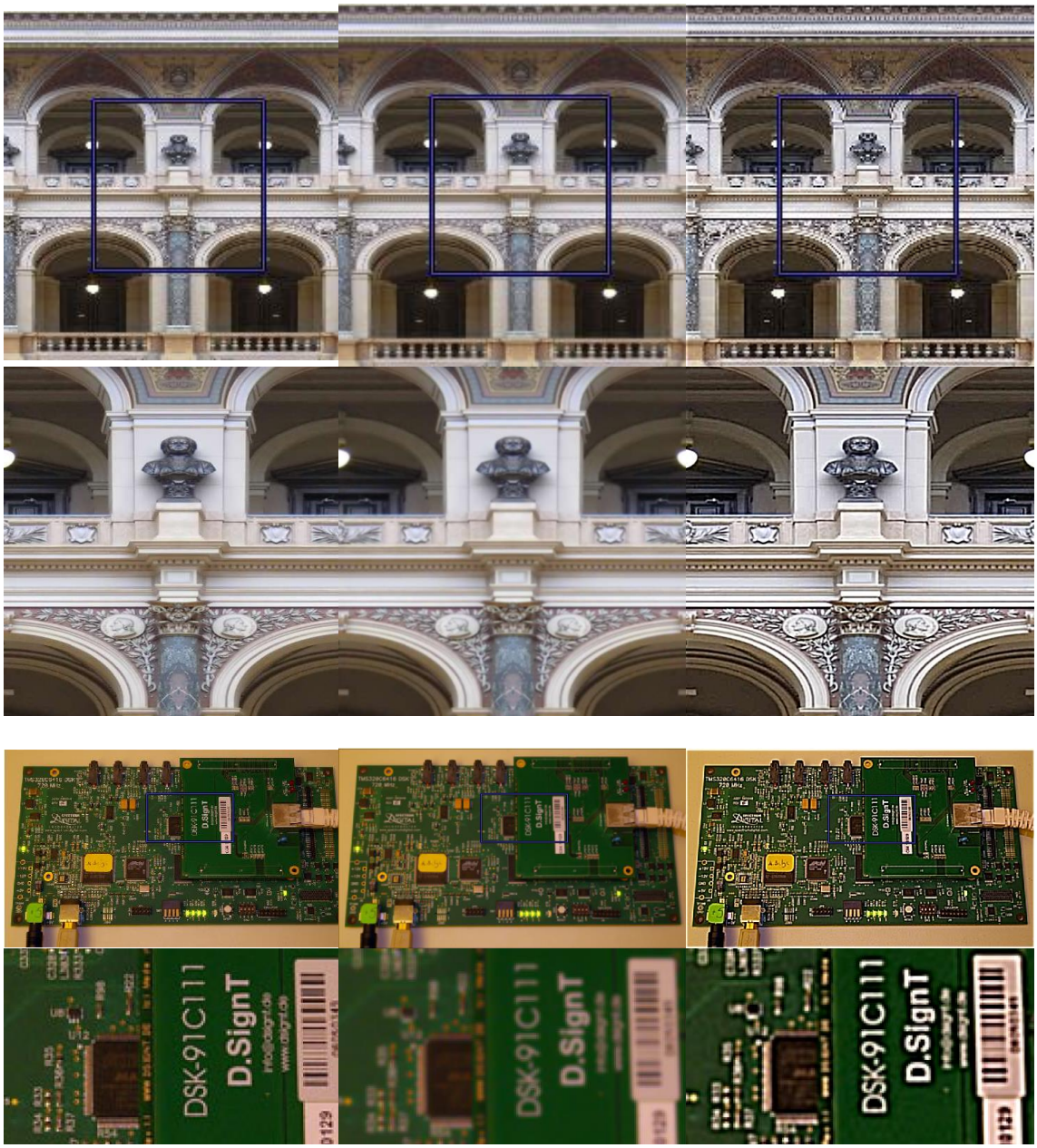
Deblurring using modified DFD2 K=0.7 and v=0.85

Original image	Blurred image	Restored image
		

Gaussian blur using sigma $\sigma = 1$ and $11x11$ kernel size

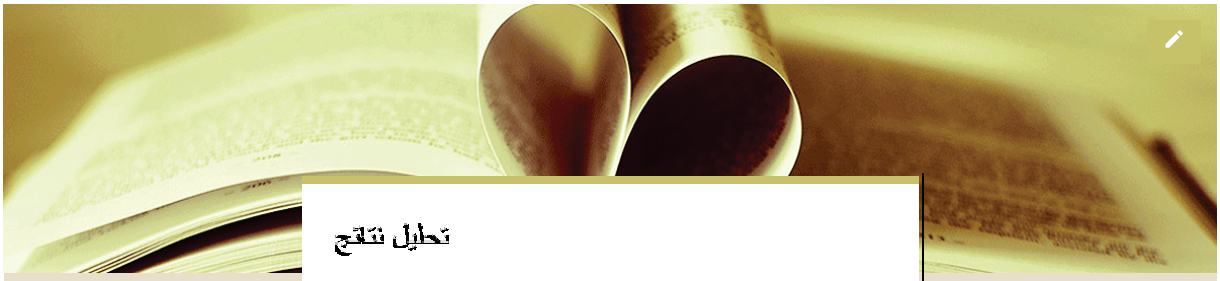
Deblurring using modified DFD2 K=0.7 and v=0.85

Original image Blurred image Restored image



Appendex-D

Samples of Google Forms



تحليل نتائج

هذا الاستبيان لتحليل بعض الصور الملتقطة برسالة الماجستير في هندسة الالكترونيات قسم هندسة الحاسوب والسلامة

* Required

Q1\ Which of these image is the best quality? افضل الصورتين أي جودة؟*

a

b

Q2\ Which of these image is the best quality? افضل الصورتين أي جودة؟*

a

b

Q3\ Which of these image is the best quality? افضل الصورتين أي جودة؟*

a

b

Q4\ Which of these image is the best quality? افضل الصورتين أي جودة؟*

a

b

Q5\ Which of these image is the best quality? افضلالصورتيزمأي جودة؟*

a

b

Q6\ Which of these image is the best quality? افضلالصورتيزمأي جودة؟*

a

b

Q7\ Which of these image is the best quality? افضلالصورتيزمأي جودة؟*

a

b

Q8\ Which of these image is the best quality? افضلالصورتيزمأي جودة؟*

a

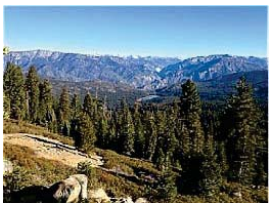
b

Q9\ Which of these image is the best quality? افضل الصورتين من أي جودة؟*

a

b

Q10\ Which of these image is the best quality? افضل الصورتين من أي جودة؟*



a

b

Q11\ Which of these image is the best quality? افضل الصورتين من أي جودة؟*

a

b

Q12\ Which of these image is the best quality? افضل الصورتين من أي جودة؟*

a

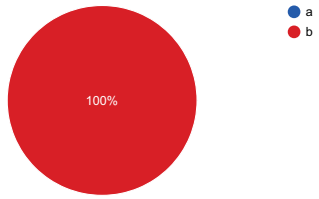
b

تحليلنتائج

125 responses

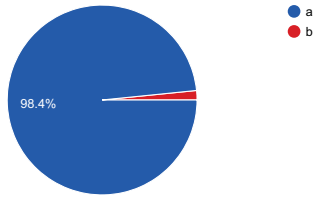
Q1\ Which of these image is the best quality? جودةأفضلالصورتينمأي

125 responses



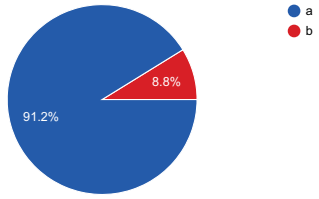
Q2\ Which of these image is the best quality? جودةأفضلالصورتينمأي

125 responses



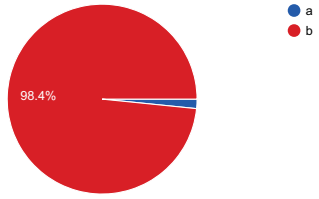
Q3\ Which of these image is the best quality? جودةأفضلالصورتينمأي

125 responses



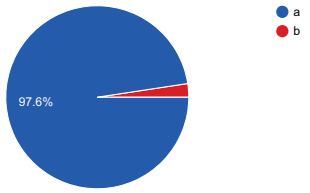
Q4\ Which of these image is the best quality? جودةأفضلالصورتينمأي

125 responses



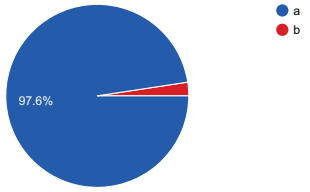
Q5\ Which of these image is the best quality? جودةأفضلالصورتينمأي

125 responses



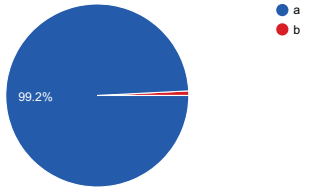
Q6\ Which of these image is the best quality? جودةأفضلالصورتينمأي

125 responses



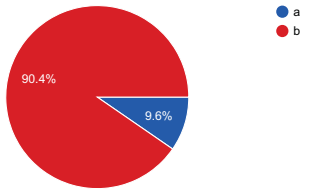
Q7\ Which of these image is the best quality? جودةأفضلالصورتينمأي

125 responses



Q8\ Which of these image is the best quality? جودةأفضلالصورتينمأي

125 responses



Q9\ Which of these image is the best quality? جودةأفضلالصورتينمأي

125 responses

الخلاصـة

لقد أصبحت المعالجة الفيديوية مؤخراً من المواضيع المهمة نتيجة للتطورات الحاصلة في تقنية المعلومات. تتناول هذه الرسالة بناء كيان مادي لخوارزمية إزالة تضبب الفيديو باستخدام مصفوفة البوابات المبرمجة حقلياً. الخوارزمية المقترحة تعتمد على التفاضل كسري المترتبة وخوارزمية فان سيترت (Van Cittert) المعدلة. تم استخدام الخوارزمية الجينية للحصول على المعاملات المثلى للخوارزمية المقترحة. إن دالة الهدف المستخدمة في الخوارزمية الجينية تعتمد على المنطق المضبب لدالتي ذروة الإشارة نسبة للضوضاء (PSNR) ومقدار العشوائية للإشارة (Entropy).

تم استخدام (Matlab 2017b) كبيئة تطوير في تصميم وبناء تطبيق مدعّم بواجهات المستخدم الرسومية (GUI) لتقيم الخوارزمية المقترحة ومقارنة نتائجها مع المرشحات صحيحة المرتبة التقليدية (Sobel, Robert, Prewitt, and Laplacian) باستخدام معايير كمية ونوعية. استعملت نماذج قوقل (Google Forms) لعمل استبيان على الانترنت لغرض التقييم البصري للنتائج. أكثر من ١٢٥ شخص شاركوا في هذا الاستبيان عبر الانترنت. ٨٨،٤% من الأشخاص المشاركين في الاستبيان اكدوا ان الصورة المسترجعة باستخدام الخوارزمية المقترحة افضل جودة من الطرائق الأخرى. كما تم استخدام (Normalized absolute error, structural content, normalized cross-correlation, peak signal to noise ratio, mean square error, entropy, average difference, and maximum difference) في التقييم الكمي للخوارزمية.

تم استخدام برمجيات شركة زايلنكس (Xilinx ISE design suite 14.5) وربطها مع الماتلاب (Matlab) في بناء الكيان المادي للخوارزمية المقترحة. تم اعتماد طريقتين في البناء، الأولى باستخدام (Xilinx system generator XSG) والأخرى باستعمال (FPGA in the loop and

ModelSim). تم التطرق لجميع جوانب تقليل المساحة المادية للتصميم على لوحة التطوير، حيث تم الحصول على نسبة تقليل ٤٤% و ٥٤% و ١٥% و ٤١% من عدد المسجلات (number of registers) و LUTs و IOB و المعالج المساعد DSP48A. وتم الوصول الى إمكانية معالجة فيديو بدقة ١٢٨x١٢٨ في الطريقة الأولى، بينما تم الوصول الى ١٠٨٠x١٩٢٠ في الطريقة الثانية في ظروف الزمن الحقيقي للتشغيل.

إقرار لجنة المناقشة

نشهد بأننا أعضاء لجنة التقويم والمناقشة قد اطلعنا على هذه الرسالة الموسومة **(بناء مصفوفة بوابات مبرمجة حقلياً لإزالة تضبب الفيديو)** وناقشنا الطالب **(عبد العزيز حسين مرعي حسن)** في محتوياتها وفيما له علاقة بها بتاريخ / /2018 وقد وجدناه جديراً بنيل شهادة الماجستير–علوم في اختصاص هندسة الحاسوب والمعلوماتية.

التوقيع: التوقيع:

رئيس اللجنة: عضو اللجنة(المشرف): د.عماد عطية خلف

التاريخ: / /2018 التاريخ: / /2018

التوقيع: التوقيع:

عضو اللجنة: عضو اللجنة (المشرف):

التاريخ: / /2018 التاريخ: / /2018

قرار مجلس الكلية

اجتمع مجلس كلية هندسة الالكترونيات بجلسته المنعقدة بتاريخ : / /2018

وقرر المجلس منح الطالب شهادة الماجستير علوم في اختصاص هندسة الحاسوب والمعلوماتية.

مقرر المجلس: **د.** رئيس مجلس الكلية:

التاريخ: / /2018 التاريخ: / /2018

جامعة الموصل

كلية هندسة الالكترونيات

بناء مصفوفة بوابات مبرمجة حقلياً لإزالة تضبب الفيديو

رسالة تقدم بها

عبد العزيز حسين مرعي حسن

إلى

مجلس كلية هندسة الالكترونيات

جامعة الموصل

كجزء من متطلبات نيل شهادة الماجستير

في

هندسة الحاسوب والمعلوماتية

بإشراف

د. عماد عطية خلف

١٤٣٩ هـ

٢٠١٨م

جامعة الموصل
كلية هندسة الالكترونيات

بناء مصفوفة بوابات مبرمجة حقليا لإزالة تضبب الفيديو

عبد العزيز حسين مرعي

رسالة ماجستير

هندسة الحاسوب والمعلوماتية

بإشراف

د. عماد عطية خلف

١٤٣٩ هـ

٢٠١٨م