Design of an Efficient Edge Enhanced Image Scalar for Image Processing Applications

[1]Rashmi. H, [2]Suganya. S
[1]PG Student [VLSI], Dept. of ECE, CMRIT, Bangalore, Karnataka, India
[2]Associate Professor, Dept. of ECE, CMRIT, Bangalore, Karnataka, India

***Abstract* - Image scaling is one of the most widely used methods in image processing applications and deals with the resizing of an image. Image scaling deals with the trade-off between efficiency, sharpness and smoothness. So this paper presents a low complexity, low memory requirement, high quality image after scaling by using an efficient edge enhanced algorithm. The proposed algorithm consists of a sharpening filter, bilinear interpolation and an edge detector. Sharpening filter is used to reduce the blurred images and sobel operator is used as edge detector to identify the edges and to reduce the noise effects in an image. And a simplified bilinear interpolation technique is used to reduce silicon cost.**

***Keywords* - Sharpening Filter, Edge detector, Bilinear Interpolation**

I. INTRODUCTION

Image scaling is a very important technique and is widely used in image processing applications. Image scaling is all about resizing of a digital image. Scaling deals with the trade off between efficiency, sharpness and smoothness. Image scaling is a non-trivial process. Scaling reduces or enlarges the size of the image by changing the number of pixels. Increasing the size of an image is called upsampling and reducing the size of an image is called downsampling.

Image scaling is used in computer graphics, digital image devices and video applications, as the use of these applications grow up, significance of image scaling becomes more outstanding.

Scaling up an image is generally done to enlarge the image, so as to make the smaller images fit into full screen mode. And scaling down is done to shrink the image, so as to make the image fit into small size LCD panel. A number of image scaling algorithms have been proposed, the most popularly used methods are nearest neighbour algorithm, bilinear interpolation algorithm and bicubic interpolation algorithm.

The image scaling algorithms are divided into two types: Polynomial and non-polynomial based method. Nearest neighbour algorithm is a polynomial based method. Scaling using bilinear and bicubic interpolation algorithms are the other polynomial based method. These methods can be easily implemented in hardware but results in blocking and blurring effects. Other non-polynomial based methods have been proposed, these methods improve the quality of image by using some of the efficient techniques such as bilinear filter, curvature interpolation. These methods improve the image quality while reducing the blocking and blurring effects. These methods provide a high quality image but have high complexity and high memory requirements.

II.PROPOSED SCALING ALGORITHM

In image scaling or resizing of an image, the sharpening filter is used to reduce the blurring effects. The edge detector is used to detect sharp discontinuities in an image. A register bank is used in this project in order to store the gray scale values of images. And bilinear interpolation method is used for image zooming. The gray scale image values are given to the memory bank, Fig 1. Shows the block diagram of the proposed scaling algorithm.

The image is first given in MATLAB, which performs image to text conversion. This code is then given as an input to Xilinx. So the sharpening filter increases the edge content. Since the edges enclose the shape information of an image, it is necessary to enhance the edges. The edge detector identifies horizontal and vertical gradient and then the gradients are added to form edges. The multiplexer provides the input to the bilinear interpolator depending on the values of sharpening filter, memory bank and edge detector. The bilinear interpolation results in blurred images which are then reduced by prefiltering it using sharpening filter. The bilinear interpolation performs scaling of an image. Finally the output image is obtained.

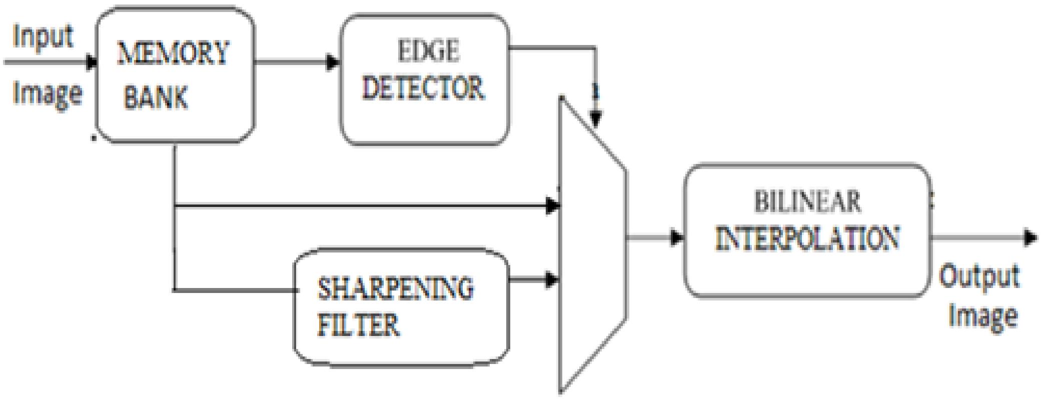


Fig 1: Block diagram of the proposed scalar algorithm

A. Sharpening Filter

Sharpening is an enhancement technique that highlights the edges , line structures and other fine details in an image Sharpening of an image consists of adding a signal to the original image , the signal that is added is proportional to high pass filter version of original image. The original image is first filtered out by high pass filter, and then the output of high pass filter is added to the original image, which thus results in a sharpened image. The choice of high pass filter is one of the key factor in sharpening process. High pass filter amplifies noise and makes the image sharper. The filters that are used for sharpening process, changes the pixel value by considering the values of neighbouring pixels.

Laplacian filter is an example of second order derivative method. Sharp discontinuity in the image such as noise will be enhanced by using this operator.

Laplacian highlights sharp discontinuities and in the output image the grey line represents the edges in an image with a black background. The laplacian operator is given by the convolution between the image and a kernel. The filtered image is added with the original image to obtain the sharpened image. Laplacian filter is a form of isotropic filter, i.e, the rotation of the filter is invariant

Laplacian kernel invariant to 90˚ rotation:

0	1	0
1	-4	1
0	1	0

Laplacian kernel invariant to 45˚ rotation:

1	1	1
1	-8	1
1	1	1

For 1D, the second order derivative is given by

$$\frac{d^2f}{dx^2} = f(x+1) + f(x-1) - 2f(x) \quad (1)$$

$$\frac{d^2f}{dy^2} = f(x,y+1) - 2f(x,y) + f(x,y-1) \quad (2)$$

The second order derivative have stronger response and are good for finding fine details in an image and are used for image enhancement. The second order derivative is more sensitive to intensity variation and the derivative depends upon the direction along which it is evaluated.

Adding both the equations, (1) & (2) we get

$$\nabla^2 f(x,y) = \frac{d^2f}{dx^2} + \frac{d^2f}{dy^2} \quad (3)$$

$$\nabla^2 f(x,y) \cong f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y) \qquad (4)$$

Many of the filtering operations or image processing operations are modelled as linear system. Convolution filter is a kind of linear filter which uses linear combination to enhance the image. The convolution kernel is a matrix of weights, which takes two functions, w and f, and produces a third function, g . It takes the weighted sum of neighbouring pixels of an image as input and obtains the output pixel value. The third function, i.e, the output represents the amount of overlap between the functions inverted mask, w and$f$. The filtered image is a convolution given by,

$$g(x,y) = w(i,j) * f(i,j) \qquad (5)$$

$$g(x,y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} w(i,j) * f(x+i,y+j) \qquad (6)$$

Each of the neighbourhood pixel is multiplied by the weight and then we add them together to obtain the output.

$$P'(m,n) = [w(m,n).P(m,n) + w(m+1,n).P(m+1,n) + w(m,n-1).P(m,n-1) + w(m,n+1).P(m,n+1) + w(m-1,n).P(m-1,n)]/[w(m,n) + w(m+1,n) + w(m,n-1) + w(m,n+1) + w(m-1,n)] \qquad (7)$$

Where, w is the weight of neighbourhood input pixel, w is rotated by 180° to form inverted mask. And P is input pixel.

B. Edge Detector

Edges usually represent object boundaries or edges are discontinuities in image with strong intensity contrast, where the pixel value changes abruptly. Edges are always important characteristics of an image. In order to emphasise more on edges in an image then some weights in the filter has to be negative. For the difference in the input pixel values, sum of sets of weights tends to produce zero output.

The sobel operator uses two 3x3 kernels, one kernel calculates the gradient in x- direction (horizontal) and other gradient in y-direction (vertical). The two kernels used are 0 deg and 90 deg convolution kernel. Each of these kernels is convolved with the original image and calculates the gradient at each point and then increases the image intensity at each point. The two kernels are

$$gx = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \&$$

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

Sobel operator is magnitude of gradient given by

$$M = \sqrt{{g_x}^2 + {g_y}^2} \qquad (8)$$

$$g_x(x,y) = -f(x-1,y-1) - 2f(x-1,y) - f(x-1,y+1) + f(x+1,y-1) + 2f(x+1,y) + f(x+1,y+1) \qquad (9)$$

$$g_y(x,y) = -f(x-1,y-1) - 2f - f(x+1,y-1) + f\,(x-1,y+1) + 2f(x,y+1) + f(x+1,y+1) \qquad (10)$$

C.Bilinear Interpolation

Bilinear interpolation method is an extension of linear interpolation. This method is opted because of its high quality and low complexity. This interpolation method, firstly it performs linear interpolation in one direction (horizontal) and then in another direction (vertical). In this method the weighted average of the nearest four pixel values is taken as input and then determines the intensity and provides an output.

The output pixel $P(k,l)$ is calculated by performing linear interpolation operation in both x and y direction. Both dx and dy are scaled parameters in horizontal and vertical directions and both of which can be specified by the user.

As shown in the below figure, a block is shown with four input pixels

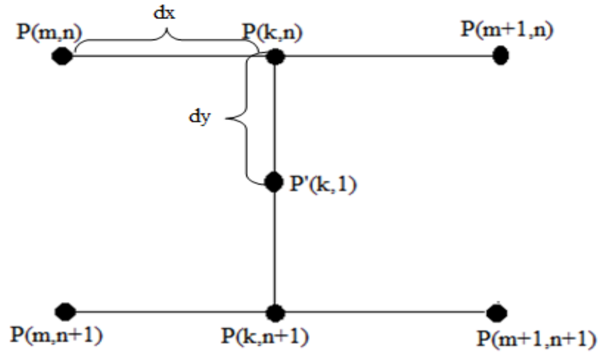


Fig 2 : Bilinear interpolation

where M is the width of the original image and N is the height .The temporary pixel values is calculated in x direction.

$$P(k,n) = (1-dx)P(m,n) + dx\,P(m+1,n) \tag{11}$$

$$P(k,n+1) = (1-dx)P(m,n+1) + dx\,P(m+1,n+1) \tag{12}$$

The output pixel value is obtained by performing linear interpolation in y direction. The output pixel $P'(k,l)$ is obtained by

$$P'(k,l) = (1-dy)P(k,n) + \ dy\,P(k,n+1) \tag{13}$$

The output pixel can be calculated by taking both horizontal and vertical interpolations and obtained by only one stage as,

$$P'(k,1) = (1-dx)(1-dy)P(m,n) + (1-dx)dy\,P(m,n+1) + dx(1-dy)P(m+1,n) + dx\,dy\,P(m+1,n+1) \tag{14}$$

dx and dy are zooming ratios in horizontal and vertical directions. It is selected because of its computational efficiency and quantitative stability. Because of its simple architecture and less complexity, it is efficient for VLSI implementation.

The computing resources of this interpolation method costs eight multiply, four subtract and three addition operations. So eight multipliers and seven adders are required for its implementation.

In order to reduce the cost of computing resources algebraic manipulation and a hardware sharing technique is used. In order to reduce the hardware cost or the silicon cost we are using a hardware sharing technique and also to reduce the delay path. The architecture of the bilinear interpolation is shown below

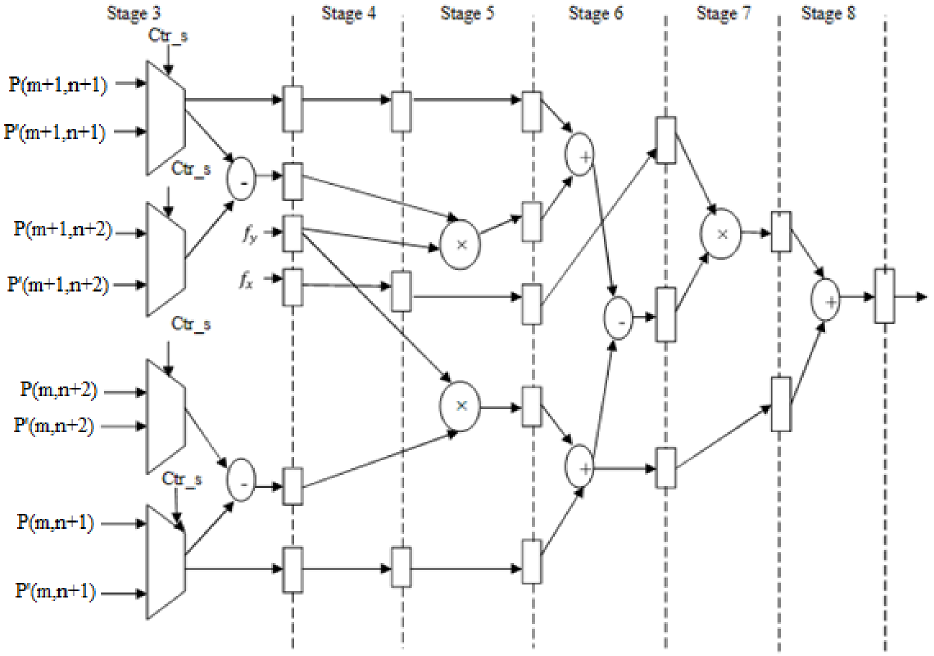


Fig 3 : Architecture of Bilinear Interpolation

The above figure, from stage 3 to stage 8 represents the bilinear interpolation architecture. It uses six stage pipeline architecture and a two stage pipelined multiplier to reduce the delay path. The input values to the multiplexer are provided by the memory bank and the sharp filter and the edge detector provides a control signal to the multiplexer. The multiplexer then selects the filtered value from the filter or the memory bank as input to the bilinear interpolation.

$$Pc(k,l) = (1-dx)(1-dy)Pb(m,n) + dx(1-dy)Pb(m+1,n) + (1-dx)dy.Pb(m,n+1) + dx.dy.Pb(m+1,n+1) \quad (15)$$

$$= dx[-(1-dy)Pb(m,n) + (1-dy)Pb(m+1,n) + dy(-Pb(m,n+1) + Pb(m+1,n+1)] + (1-dy)Pb(m,n) + dy.Pb(m,n+1) \quad (16)$$

$$= dx\{dy[Pb(m+1,n+1) - Pb(m,n+1) - Pb(m+1,n)Pb(m,n)] + Pb(m+1,n) - Pb(m,n)\} + dy[Pb(m,n+1) - Pb(m,n)] + Pb(m,n) \quad (17)$$

$$= dx\{dy[Pb(m+1,n+1) - Pb(m+1,n)] + Pb(m+1,n) - dy(Pb(m,n+1) - Pb(m,n) + Pb(m,n)\} + dy(Pb(m,n+1) - Pb(m,n) + Pb(m,n) \quad (18)$$

The output obtained from bilinear interpolation, i.e, eqn (15) can be simplified by algebraic manipulation. The equation (15) contains $dy.Pb(m,n+1) - Pb(m,n)$ twice, so one of the calculation function can be reduced. So by addition of three registers the function $dy.Pb(m,n+1) - Pb(m,n)$ can be reduced by the hardware sharing technique. The calculation function is replaced; i.e, one multiplexer and two adders are reduced. Since the values of dx & dy are floating point numbers from 0 to 1 and the pixel values are integer from 0 to 255. To reduce the computation cost, low cost integer calculation is done. Firstly, we shift dx & dy left by 8bits, so that the values of dx & dy are multiplied by 256. After this multiplication the highest 8bits obtained are approximated integer values of equation. The target result $Pc(k,l)$ is computed by low cost integer addition, subtraction and multiplication. So by this approximation technique the floating point computation is replaced by integer computation.

.

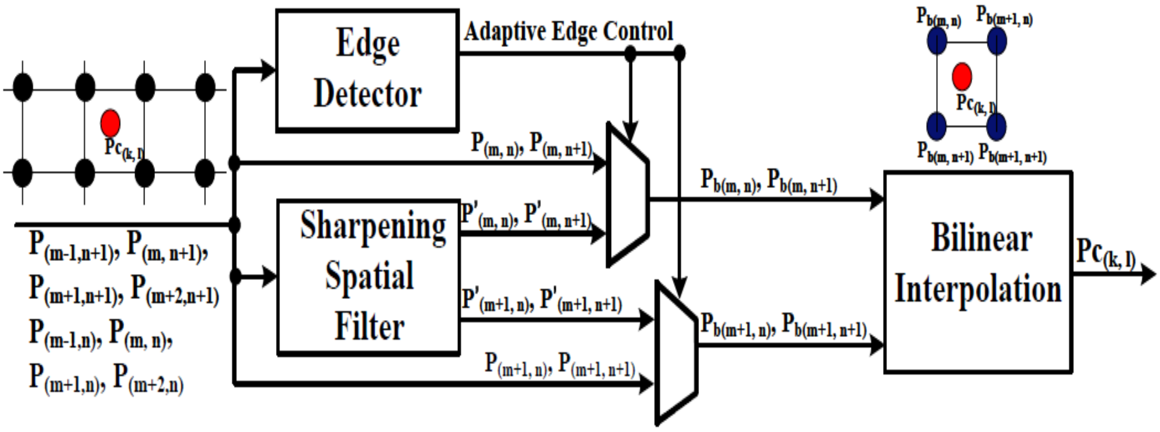

Edge Detector
Adaptive Edge Control
$P_{(m,n)}, P_{(m,n+1)}$
Sharpening Spatial Filter
$P'_{(m,n)}, P'_{(m,n+1)}$
$P'_{(m+1,n)}, P'_{(m+1,n+1)}$
$P_{(m+1,n)}, P_{(m+1,n+1)}$
$P_{b(m,n)}, P_{b(m,n+1)}$
$P_{b(m+1,n)}, P_{b(m+1,n+1)}$
Bilinear Interpolation
$Pc_{(k,l)}$
$Pc_{(k,l)}$
$P_{(m-1,n+1)}, P_{(m,n+1)},$
$P_{(m+1,n+1)}, P_{(m+2,n+1)}$
$P_{(m-1,n)}, P_{(m,n)},$
$P_{(m+1,n)}, P_{(m+2,n)}$
$P_{b(m,n)}$ $P_{b(m+1,n)}$
$Pc_{(k,l)}$
$P_{b(m,n+1)}$ $P_{b(m+1,n+1)}$

Fig 4: Block diagram of the edge enhanced scaling algorithm

III.SIMULATION RESULTS

The MATLAB tool is used to evaluate the performance of the prposed scalar architecture. Xilinix ISE tool is used for synthesis of the design and SPARTEN 3E is used for FPGA implementation.

Image output after scaling up of image:

The experimental results for scaling up of image in MATLAB are shown in the below figure.

Fig 5:(a)Shows input image & (b) Shows output image & fig (c) shows the simulation waveform

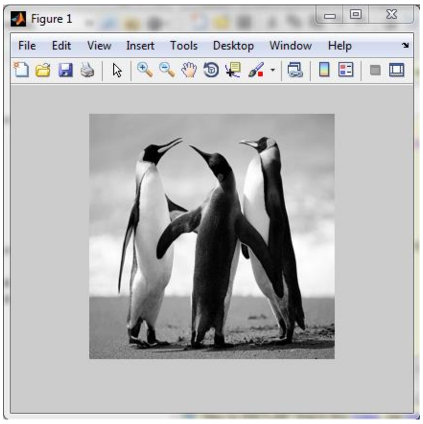


Fig 5(a): Input image

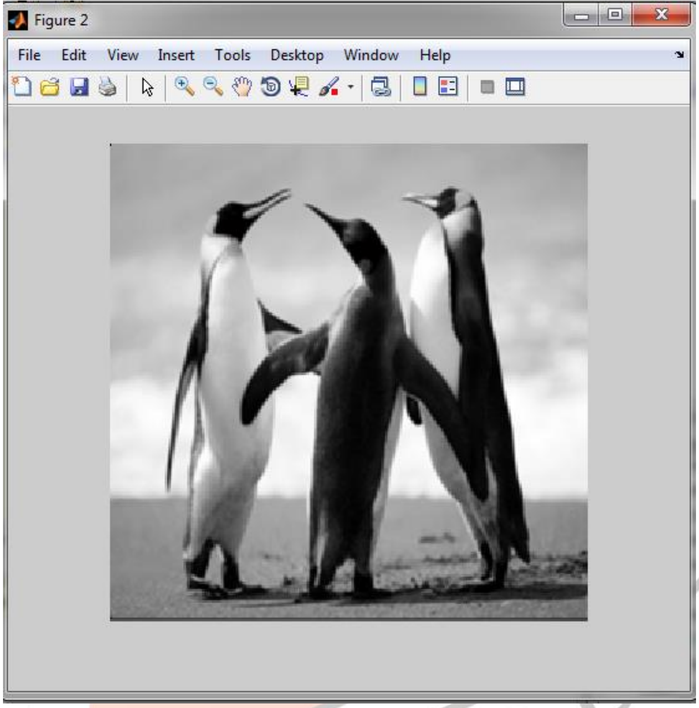
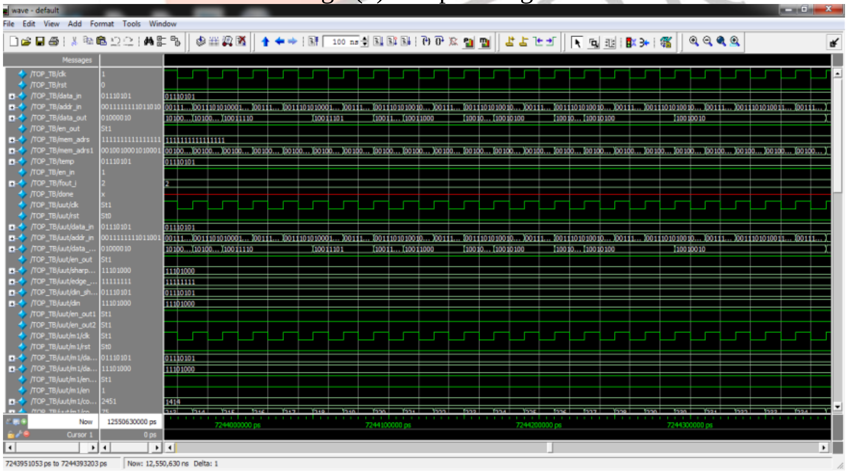


Fig 5(b): Output image



```
/TOP_TB/clk              1
/TOP_TB/rst              0
/TOP_TB/data_in          01110101
/TOP_TB/addr_in          0011111111011010
/TOP_TB/data_out         01000010
/TOP_TB/en_out           St1
/TOP_TB/mem_adrs         1111111111111111
/TOP_TB/mem_adrs1        0010010001010001
/TOP_TB/temp             01110101
/TOP_TB/en_in            1
/TOP_TB/fout_i           2
/TOP_TB/done             x
/TOP_TB/uut/clk          St1
/TOP_TB/uut/rst          St0
/TOP_TB/uut/data_in      01110101
/TOP_TB/uut/addr_in      0011111111011001
/TOP_TB/uut/data_...     01000010
/TOP_TB/uut/en_out       St1
/TOP_TB/uut/sharp...     11101000
/TOP_TB/uut/edge_...     11111111
/TOP_TB/uut/din_sh...    01110101
/TOP_TB/uut/din          11101000
/TOP_TB/uut/en_out1      St1
/TOP_TB/uut/en_out2      St1
/TOP_TB/uut/m1/clk       St1
/TOP_TB/uut/m1/rst       St0
/TOP_TB/uut/m1/da...     01110101
/TOP_TB/uut/m1/da...     11101000
/TOP_TB/uut/m1/en...     St1
/TOP_TB/uut/m1/en        1
/TOP_TB/uut/m1/co...     2451

Now       12550630000 ps
Cursor 1  0 ps

7243951053 ps to 7244393203 ps    Now: 12,550,630 ns  Delta: 1
```

Fig 5(c): Fig shows simulation waveform

6. Image output after scaling down of image:

The experimental results for scaling down of image in MATLAB are shown in below figure.

Fig (a) Shows input image ,Fig (b) Shows output image & fig (c) shows the simulation waveform

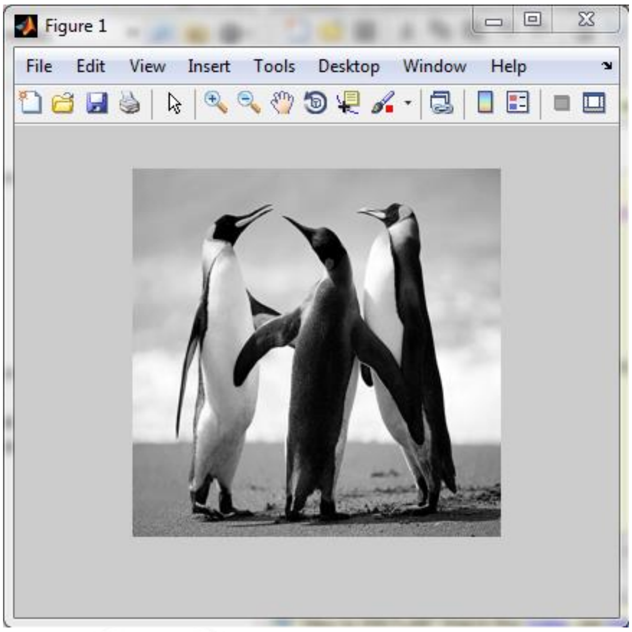


Fig 6(a) : Input image

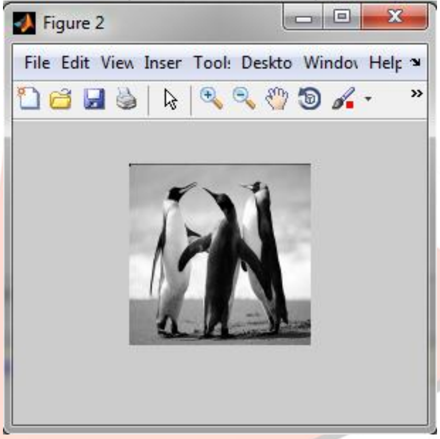


Fig 6(b) : Output image

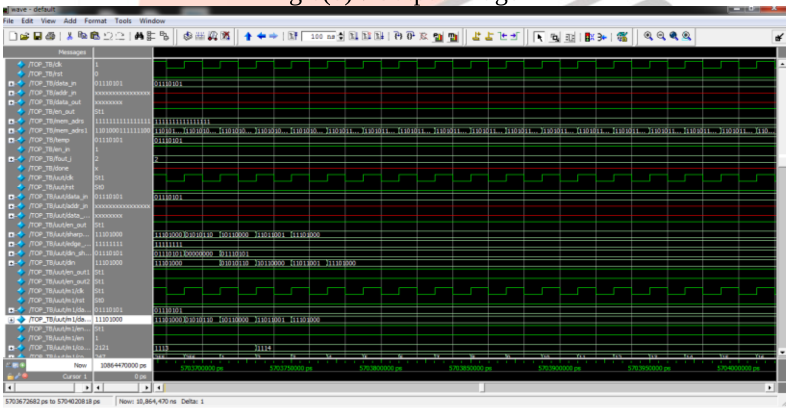


Fig 6(c): Fig shows simulation waveform

IV.CONCLUSION

In this paper a low cost, low memory requirement, low complexity , high quality and high performance scalar architecture is proposed..In this paper a convolution filter and sobel operator are introduced to reduce blurred images and the presence of noise in image is also reduced to some extent. And hardware sharing technique for bilinear interpolation is used in order to reduce the silicon cost. The design was simulated by using the simulator Modelsim SE 6.3f and synthesised using Xilinx ISE 12.2. MATLAB tool is used to obtain the output image.

ACKNOWLEDGMENT

We thank the referees for their suggestions to improve the content of this paper. The authors would thank the Principal for providing necessary facilities to carry out the work. We would thank our Head of the Department for guiding us towards the implementation.

REFERENCES

[1] Shin–Lun Chen “VLSI Implementation of an Adaptive Edge-Enhanced Image Scalar for Real –Time Multimedia application” IEEE Trans. circuits and systems for video technology,2013.

[2] Shin–Lun Chen “VLSI implementation of a low-Cost High-quality Image Scaling Processor”, IEEE Trans. circuits and system, vol.60, no.1, Jan.2013 .

[3] C. C. Huang, P. Y. Chen, and C. H. Ma, “A novel interpolation chip for real-time multimedia application,” IEEE Transaction on Circuits and Systems for Video Technology, Vol. 22, no. 10 , pp. 1512-1525, Oct. 2012.

[4] Christos Gentsos, Calliope-Louisa Sotiropoulou and Spiridon Nikolaidis “Real- Time Canny Edge Detection Parallel Implementation for FPGAs” in Electronics, Circuits, and systems (ICECS), 2010 17th IEEE International Conference.

[5] Fahad Alzahrani , Tom Chen “A Real-Time High Performance Edge Detector for Computer Vision Applications ” in Proc. of Design Automation Conference, 1997. Proceedings of the ASP-DAC '97 Asia and South Pacific .

[6] Chung-Chi Lin, Ming-Hwa Sheu, Huann-Keng Chiang , Chishyan Lia Zeng-Chaun Wu and Wen-Kai Tsai “An Efficient Architecture of Extended Linear Interpolation for Image Processing ” in Journal of Information Science And Engineering 26, 631-648 (2010)

[7] R.Sundar “An Efficient Low cost Image scaling Technique for less power Consumption ” in International Journal of Advance Research in Computer Science and Management Studies Volume 2, Issue 2, February 2014.

[8]Ms. Abhilasha Bhatnagar , Dr.R.Ramakishore “Spline based Interpolation methods For Image Magnification ” in International Journal of Scientific & Engineering Research, Volume 4, Issue 5, May-2013.

[9]Robert G..Keys “Cubic Convolution Interpolation for Digital Image Processing” in IEEE Transactions on Acoustics, Speech , and Signal Processing, VOL. ASSP-29, NO. 6, DECEMBER 1981

[10] Shih-Lun Chen, Hong-Yi Huang and Ching-Hsing Luo, “A Low-Cost High-Quality Adaptive Scalar for Real-Time Multimedia Applications” in IEEE Transactions on circuits and systems for video technology, VOL 21, NO. 11, November 2011.

[11] G.Manoj kannan,A.Manoj kumar,R.Mareeswaran, J.Kanimozhi “VLSI Architecture of Pipelined Adaptive Edge-Enhanced Image Scalar for Image Processing Applications” in International Journal of Innovative Research in Science, Engineering and Technology Volume 3, Special Issue 3, March 2014

[12] M.Saranya, V.Meenakshi, “VLSI Based Image Zooming Application by a Novel Adaptive Edge Enhancement Technique”, in International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, Vol. 3, Issue 4, April 2014

[13] Mr. Harvinder Singh, Prof (Dr). J.S. Sodhi, “Image Enhancement using Sharpen Filters” , in International Journal of Latest Trends in Engineering and Technology , Vol. 2 Issue 2 March 2013