An Efficient Implementation of Image Mipmapping using Frequency Domain Techniques

Chunduri SreenivasaRao [#1], Babusa Dudekula [#2], Sathyasri Nukala [#3]
[#] AMD India Pvt Ltd, Madhapur, Hyderabad, INDIA-500081
[1] Srinivasa-Rao.Chunduri@amd.com
[2] Babusa.Dudekula@amd.com
[3] Sathya-sri.Nukala@amd.com

***Abstract*—Mipmapping is a popular technique, generally, used in real time processors to increase rendering speed and minimize aliasing effects. The basic idea of this technique is to construct a pyramid of binary fractioned downscaled images of the original image. The existing time domain convolution approach suffers from more computations for downscaling of images, particularly when the image resolution is very high. In this paper, frequency domain based overlap save method along with basic FFT properties is explained on how to implement mipmaps efficiently, aiming at less computations. The theoretical computations were provided of the proposed approach. The proposed approach was implemented on ADSP-BF533 DSP processor for different resolutions of input images. The computational comparison clearly reveals that the proposed approach is attractive to be used in scaling at very high resolution images and provides good performance than time domain filtering. The experimental results also show that proposed approach does not degrade the quality of the downscaled images based on the measurement of PSNR between proposed and time domain techniques.**

Keyword- Mipmapping, Video scaling, Convolution, Overlap Save Method

I. INTRODUCTION

Image downscaling is a fundamental operation performed constantly in digital imaging. The abundance of high resolution capture devices and the variety of displays with different resolutions make it an essential component of any application that involves images or video. In dynamic scenes, textured objects can be viewed at different distances from the viewpoint. The size of the texture along with that of projected image decreases continuously as the textured object moves farther from the viewpoint and vice versa [1]. To meet these needs, it is necessary to filter the texture map down to an appropriate size for mapping onto the object, without compromising for any visually disturbing artefacts. If high resolution is used, aliasing is increased but image looks to be very clear and if the lower resolution is used, image is overly blurry. Obviously there must be some trade-off for aliasing vs blurriness. To avoid such artefacts, mipmapping generates a pyramid of textures in which a series of pre-filtered textures map into half-decreasing resolutions at each successive level with original image resolution at pyramid level as shown in Fig.1. Mipmapping (from the Latin phrase *multum in parvo* meaning "much in little" [2]) is the process of pre-filtering one large image into many smaller images of progressively decreasing resolutions. Usually, powers of 2 are used for the mipmap levels. For example, if the highest resolution was 1024 x 1024, the next levels would be 512 x 512, 256 x 256, 128 x 128, sometimes all the way down to 8 x 8 or 1 x 1 images. The additional memory cost of mipmapping to store pre-scaled images, apart from the memory needed to store original image, is obtained by 1/4 + 1/16 + 1/64 + …= 1/3 times that of original image.

For obtaining pre-scaled images at each stage, an anti-aliasing filter must be applied to prevent aliasing. An anti-aliasing filter is a low-pass filter which can suppress high-frequency components and reduce undesirable artefacts [3]-[7]. The block diagram for decimation process is shown in Fig.2. Each mipmap level can be obtained by varying the value of D with increasing power of 2. For each mipmap level, the filter order should be doubled to preserve more image details in the downscaled image [1] [8].

This paper is organized as follows. Section 2 explains the existing techniques for mipmapping. Section 3 gives full details of proposed approach and how to optimize computation cycles with this approach. Section 4 provides the experimental results. The computational complexity of both the time-domain and frequency-domain implementation is compared in this section. Finally section 5 provides the conclusion and future scope to update the proposed method.

II. REVIEW OF PREVIOUS WORK

In literature, several techniques are available to obtain downsampled images in mipmapping. The aim of mipmapping is to pick-up the correct pixel colour among different neighbouring pixels. The easiest approach is nearest neighbour interpolation in which the closest pixel colour will be used among the closest four pixels as shown in Fig.3(a). Here the interpolated pixel color should be identified at co-ordinate (s, t). The four closest pixels are available at (s_0, t_0), (s_0, t_1), (s_1, t_0) and (s_1, t_1). Among these, the colour of pixel at (s_1, t_0) will be selected as interpolated pixel colour at co-ordinate at (s, t) because (s_1, t_0) is near to (s, t). Though this approach is very simple, the downsampled image suffers from low quality and aliasing [9]-[10].

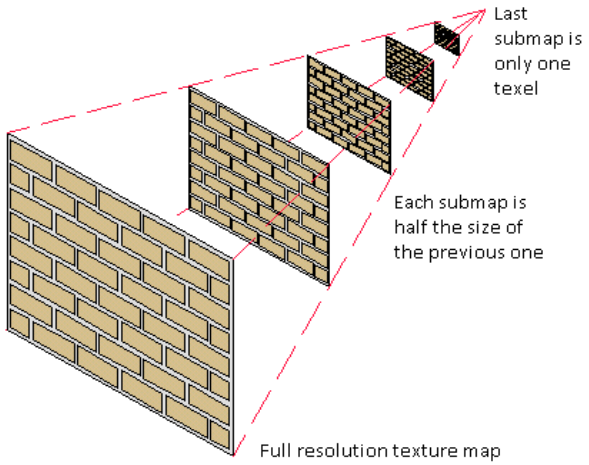


Fig. 1. An example of mipmapping pyramid textures

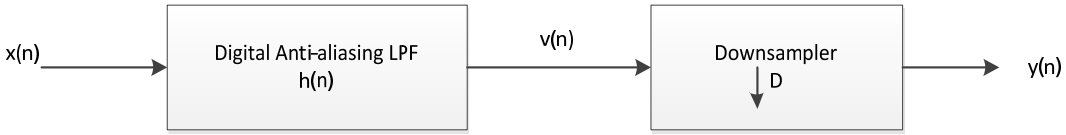


Fig. 2. Block diagram of downsampling by a factor D

Another approach is bilinear interpolation [9] in which linear interpolation is applied horizontally and vertically to obtain the desired pixel from the neighbouring pixels as shown in Fig.3 (b) and (c) respectively. Referring to Fig.3 (b), the pixels C_{top} and C_{bot} are obtained using horizontal linear interpolation as

$$C_{top} = tex(s_0, t_1)(1 - r_s) + tex(s_1, t_1) r_s \quad (1)$$

$$C_{bot} = tex(s_0, t_0)(1 - r_s) + tex(s_1, t_0) r_s \quad (2)$$

where

$tex(i,j)$ is the pixel value at the texel co-ordinate (i, j)

r_s is the pixel ratio in s-direction, which is given by

$$r_s = \frac{s - s_0}{s_1 - s_0} \quad (3)$$

Now the pixel value at (s, t) is obtained from C_{top} and C_{bot} using vertical linear interpolation (refer to Fig. 3(c)) as

$$C = C_{bot}(1 - r_t) + C_{top} r_t \quad (4)$$

where

r_t is the pixel ratio in t-direction, which is given by

$$r_t = \frac{t - t_0}{t_1 - t_0} \quad (5)$$

But bilinear interpolation is not attractive in all cases because the very small downscaled image causes accuracy problems due to missed pixels and this, in turn, results into blurriness that can lead to loss in smoothness in downscaled image [10]. An update of bilinear interpolation is trilinear interpolation in which the bilinear interpolation followed by linear interpolation is applied on the neighboring pixels to yield the desired

pixel value. But this technique can often combine pixels from outside of the sample area, particularly when the samples are of not a square type, which is not correct [11].

To overcome these issues, the original image is filtered with low-pass filter (LPF) first and then downsampled (Fig.2). To avoid aliasing effects, either Gaussian or sinc function is used as LPF [12]. Convolving the original image with discrete sinc function is computationally expensive task in time domain due to huge number of multiplications and additions. This becomes severe issue when the original image has high resolution. To obtain same output quality as that of time domain convolution without consuming too many computations, efficient implementation techniques are desired. The proposed approach is based on principle that time domain convolution is equivalent to frequency domain multiplication. The overlap save method is used for convolution process, which is efficient, particularly at high resolutions. The mathematical background of using overlap save approach for downscaling is explained and implementation details were provided in this paper. The computational comparison was provided between time domain and frequency domain based approaches.

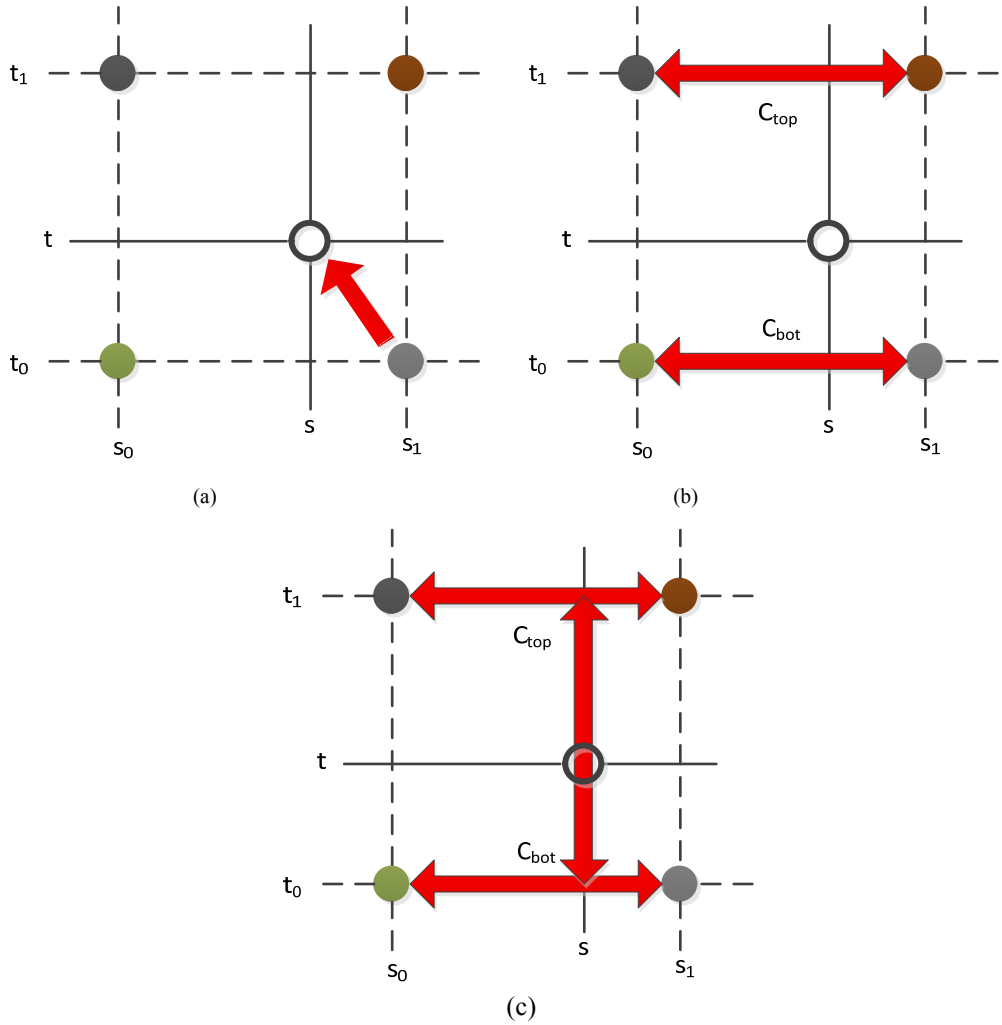


Fig. 3. (a) Nearest neighbour interpolation (b) Horizontal linear and (c) Vertical linear interpolation

III.PROPOSED IMPLEMENTATION APPROACH FOR MIPMAPPING

Referring to Fig.2, let *X(k), V(k)* and *Y(k)* be the discrete frequency equivalents of time domain signals *x(n), v(n)* and *y(n)* respectively. Referring to [5], frequency domain outputs can be expressed, in terms of continuous frequency variable, as

$$V(\omega_x) = H(\omega_x)X(\omega_x) \tag{6}$$

$$Y(\omega_y) = \frac{1}{D}\sum_{i=0}^{D-1} V\left(\frac{\omega_x - 2\pi i}{D}\right) \tag{7}$$

where ω_x and ω_y are the continuous frequency variables of *x(n)* and *y(n)* respectively and D is downsampling factor. The discrete frequency equivalents of equations (6) and (7) are given as

$$V(k) = H(k)X(k) \tag{8}$$

$$Y(k) = \frac{1}{D}\sum_{i=0}^{D-1} V\left(k + \frac{Ni}{D}\right) \tag{9}$$

where k is discrete frequency index and N is FFT length. The discrete frequency of *x(n)* varies from 0 to *N-1* when continuous time frequency ω_x is varying from 0 to 2π. In similar manner, the discrete frequency index, *k* varies from 0 to *N/D* – 1 as output frequency ω_y is varying from 0 to *2 π /D*. From equation (9), it is understood that the FFT of y(n) is sum of downsampled component, *V(k), k=0,1,…N/D – 1* and aliased components from *V(k+N/D)* to *V(k+(D-1)N/D)*. The output *Y(k)* is fed to IFFT as input and the IFFT output is scaled by the factor *1/D*. Fig.4 shows this process in block diagram.

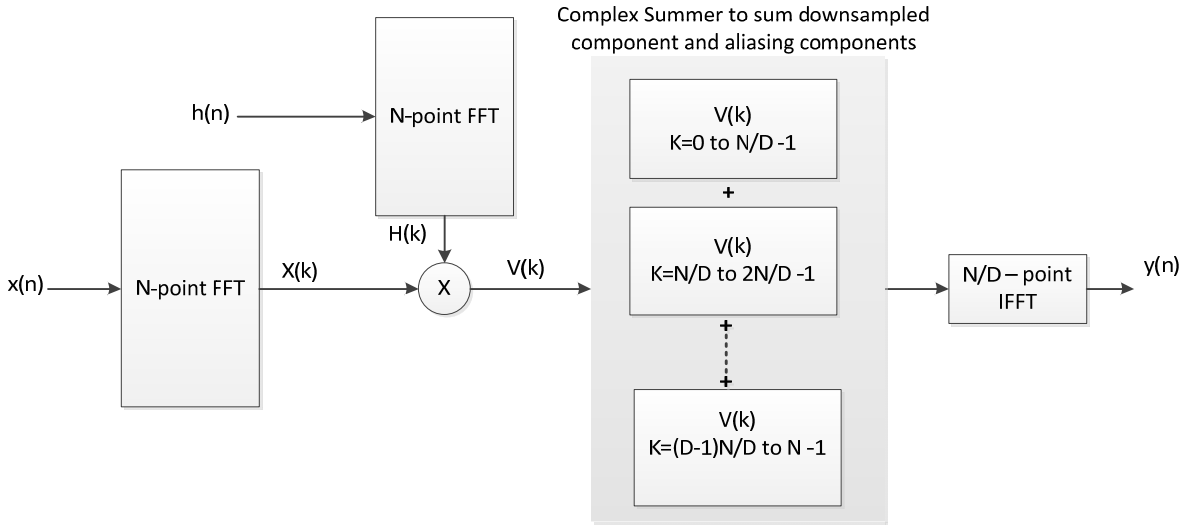


Fig. 4. Frequency Domain Downsampling Process

A. Implementation for Downsampling Original Image

The implementation procedure is explained in flow chart of Fig.5. Initially, a complex sequence is formed with two successive rows of original image. The FFT length is decided based on *N = L+M-1*, where *L* and *M* are the lengths of row and filter response respectively. If *N* is not a power of 2, sufficient zeros will be appended to complex sequence. The FFT, *H(k)* of filter response is calculated once and will be reused for every row or column processing. The FFT, *X(k)* of input complex sequence is evaluated first and complex multiplication is performed for each frequency index. From the output of complex frequency multiplication, actual downsampled component and other aliasing components are separated. All these components are summed to get the final FFT, *Y(k)* of downsampled component of *x(n)* by the downsampling factor *D*. Since the downsampled component has *N/D* frequency points, IFFT length should become *N/D*. The component, *Y(k)* is fed to IFFT block of length *N/D* and the output of IFFT is scaled with factor *1/D*. From the output *y(n)*, the desired pixels are copied to output buffer by separating real and imaginary components as two individual rows. This process will be continued for rest of rows. Also after completion of downsampling all rows, the same approach is repeated for all columns to obtain the final downsampled image with factor *D*. This process is common for obtaining downsampled image at different mipmap levels. One should note that the overlap save approach introduces a pixel delay equal to filter length, in worst case. While copying the IFFT output, one should ensure to leave the initial M pixels due to delay as they won't contain desired information.

B. Computational Complexity of Overlap Save based Downsampling

Table I provides computational complexity details. Here *N = L+M-1* and *D* is the downsampling factor. *L* and *M* are lengths of row/column and filter respectively. The computational complexity details described in above table are valid if decimation factor is a power of 2. Also computational complexity in this table is valid when two rows/columns are processed simultaneously by forming a complex sequence. The overall computational complexity will vary based on original image width and height accordingly.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

To prove the efficiency of proposed method, several images of different resolutions, such as 1920x1080, 800x533 and 640x480, were taken as input images and were applied to proposed method. The 128 taps sinc filter is used as LPF, designed using elliptic filter approach. At each mipmap level, FFT length is decided based on *N=L+M-1*. If *N* is not power of 2, sufficient zeros are padded to the row or column sequence. During processing, a complex sequence was formed with either successive rows or columns before feeding as input to

overlap save approach. This is basically done to save computations and 50\% performance improvement can be obtained with this approach.

The Analog Devices BF533 Ez-Kit Lite [13]-[14] is taken as DSP platform to implement the proposed approach due to high performance, power efficient processor architectural features as mentioned below

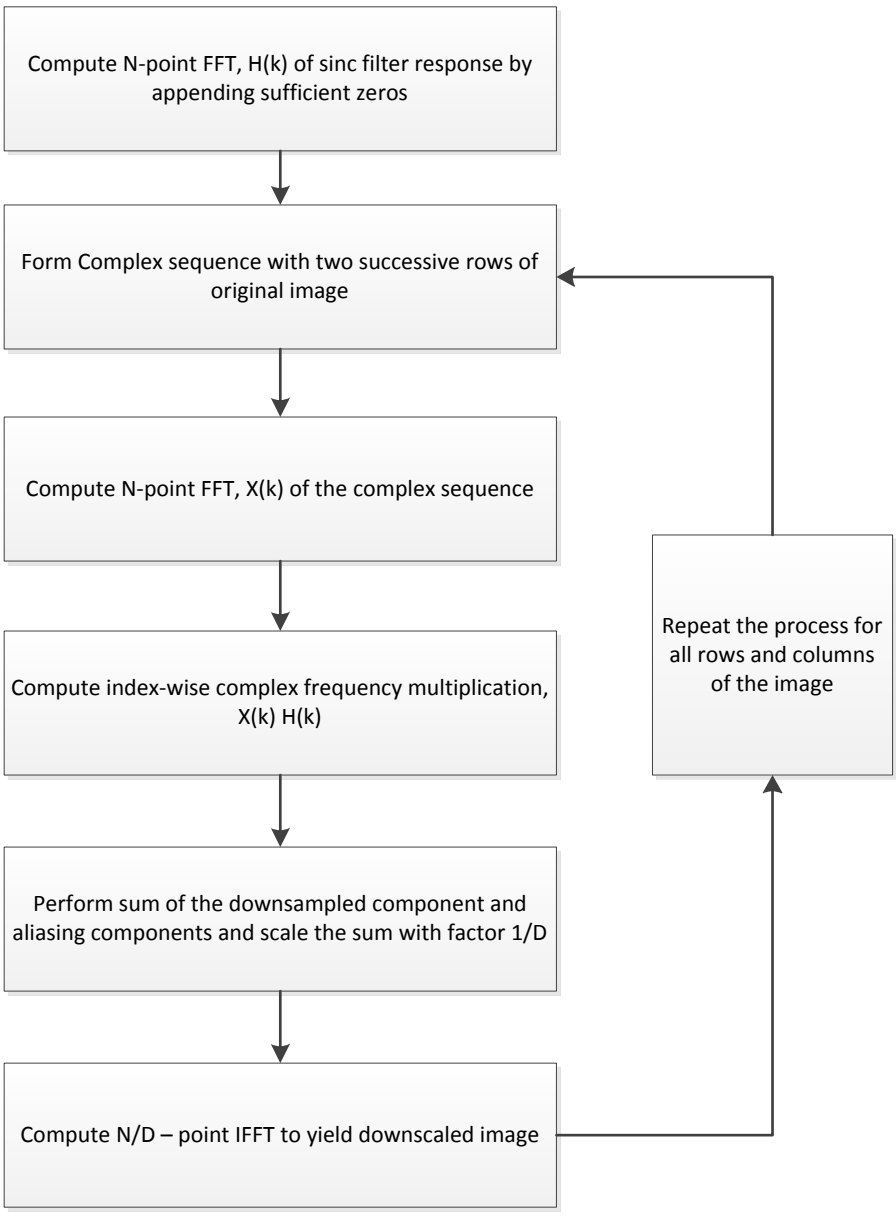


Fig. 5. Flow chart to perform filtering and downscaling of original image in mipmapping

TABLE I. Theoretical Computational Complexity of downsampling a single complex row or column using Overlap Save Approach

Component	Complex Multiplications	Complex Additions	Reamrks
X(k)	0.5 O(N)	O(N)	Complexity for calculating X(k) for complex sequence formed with either two rows or two columns using N-point FFT
V(k)	N	-	Index-wise complex frequency multiplication
Y(k)	-	N(D-1)/D	Complex frequency addition of downsampled component and aliasing components
y(n)	0.5 O(N/D)	O(N/D)	IFFT calculation of Y(k). Here IFFT length is N/D
Total	N+0.5[O(N)+O(N/D)	N(D-1)/D+ O(N)+O(N/D)	Total computations to obtain the final output y(n)

- High performance (Up to 756 MHz) 16-bit/32-bit embedded processor core
- 10-stage RISC MCU/DSP pipeline with mixed 16-bit/32-bit ISA for optimal code density
- Full SIMD architecture, including instructions for accelerated video and image processing.
- Memory controller providing glueless connection to multiple banks of external SDRAM, SRAM, Flash, or ROM
- Large on-chip SRAM for maximum system performance
- Clean, orthogonal RISC like microprocessor instruction set
- Flexible instructions to process 8, 16 and 32 bits in a register

Also, Visual DSP++ Software environment tool [15] is used for development of the proposed method. The image data was stored in external memory, SDRAM initially and rows/columns are copied into on-chip DSP RAM based on processing needs. The original image was downscaled to various mipmap levels ranging from 2 to 128. The computations are measured in terms of cycle count for the proposed approach at each mipmap level. The same measurement was done for time domain filtering approach at all mipmap levels. Table II provides the computations taken by the proposed approach and time domain filtering for image resolutions of 1920x1080, 800x533 and 640x480 respectively. The computations comparison between time-domain filtering and proposed approach was provided in Fig.6 (A), (B) and (C) respectively for the above resolutions. From the comparison, it is clear that the proposed approach yields better performance at the low decimation factors i.e. at high resolutions and the deviation in performance between TDF and OS becomes reduced as decimation factor is increasing. At high decimation factors, TDF performs better than OS approach (indicated with coloured cells in table II). The reason for this is that FFT size becomes more at high decimation factors for processing of lower number of pixels. The cycles that are required for OS approach in this case are more than that of TDF.

TABLE II. Comparison of Computations for proposed approach (OS) and time Domain Filtering (TDF) at resolutions 1920x1080, 800x533, 640x480

Decimation factor, D	**1920x1080**		**800x533**		**640x480**	
	OS	**TDF**	**OS**	**TDF**	**OS**	**TDF**
2	55.928	99.533	11.814	20.423	10.15	14.745
4	20.889	24.883	4.546	5.107	2.35	3.686
8	4.832	6.22	1.059	1.277	0.812	0.922
16	1.498	1.556	0.25	0.319	0.219	0.23
32	0.354	0.392	0.0789	0.082	0.067	0.058
64	0.086	0.098	0.019	0.024	0.016	0.014
128	0.022	0.024	0.009	0.005	0.0083	0.0036

The output quality of the proposed approach is measured by finding the PSNR (Peak Signal Noise Ratio) between output images obtained from TDF and OS methods. The output downscaled images are shown in Fig.7, 8 and 9 for input images of 1920x1080, 800x533 and 640x480 resolutions respectively. In each figure, (A), (B) and (C) contain the original image, frequency domain processed output image and time domain processed output images respectively. The PSNR in dB for all downscaled images was found out to be $-\infty$, which is clear that there is no quality difference between TDF and OS approach. At the same time, OS approach performs better than TDF at high resolutions for mipmapping.

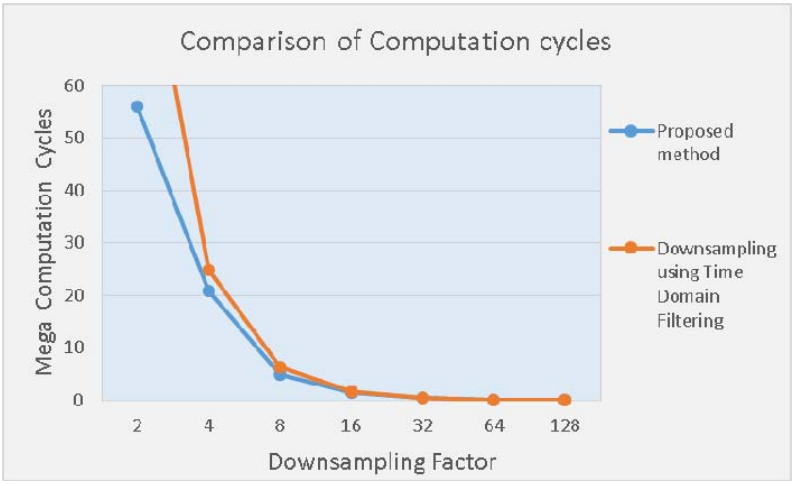
V. CONCLUSION

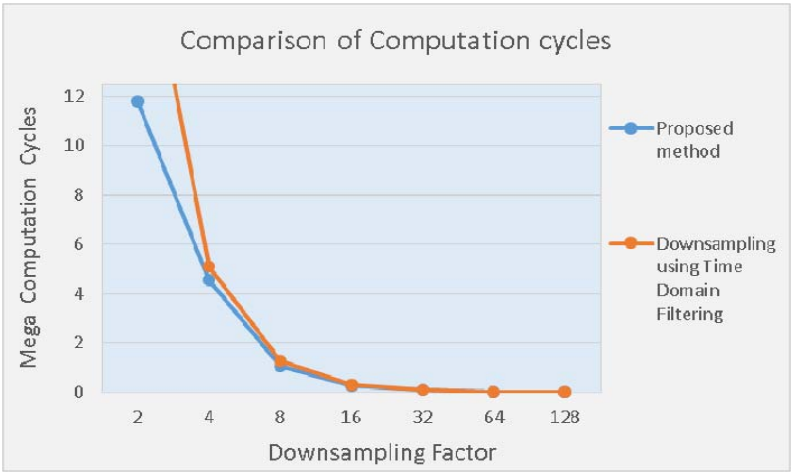
An efficient implementation approach called overlap save method is proposed to generate downscaled images in mipmapping application. This approach has the benefit of better performance over the existing time domain filtering approach, particularly at high resolutions. To prove the concept, this method was implemented on ADSP-BF533 processor for some of the input images. The results clearly show that proposed implementation approach is very efficient at high resolutions and there won't be any deviation in output quality of downscaled images.

REFERENCES

[1] Texture Filtering [Online] Available : https://elementalray.wordpress.com/tag/texture/
[2] Williams, L. “Pyramidal parametrics In Computer Graphics”, SIGGRAPH ’83 Proceedings, Vol. 17, pp. 1–11, Jul 1983.
[3] Crochiere and Rabiner, “Interpolation and decimation of signals”, Proceedings of IEEE, Vol. 69, No.3, Mar 1981.
[4] R. W. Schafer and L. R. Rabiner, “A digital signal processing approach to interpolation”, Proc. IEEE, vol. 61, pp. 692–702, 1973.
[5] John G. Proakis and Dimitris G. Manolakis, Digital Signal Processing Principles, Algorithms and Applications, 3rd edition, Prentice-Hall International Publications, New Jersey, USA, 2004
[6] Sophocles J. Orfanidis, Introduction to Signal Processing, 1st edition, Pearson Education Publications, 2010
[7] Steven W. Smith, The Scientist and Engineer’s Guide to Digital Signal Processing, California Technical Publications, 1st edition, San Diego, 2002

[8] Lindeberg, T. and Bretzner, L. Real Lindeberg, T. and Bretzner, L. "Real-time scale selection in hybrid multi time scale selection in hybrid multi-scale representations", Proc. Scale-Space'03, Isle of Skye, Scotland, Springer Lecture Notes in Computer Science, Vol.2695, pp 148-163, 2003.
[9] Prashanth.H.S, Dr Shashidhara HL, Dr KNB Murthy, "Comparative analysis of different interpolation schemes in image processing", International Conference on Advanced Communication Systems, GCT, Coimbatore , Jan 10 - 12, 2007.
[10] Jency Titus, Sebastian Geroge, "A Comparison Study On Different Interpolation Methods Based On Satellite Images", International Journal of Engineering Research & Technology, ISSN: 2278-0181, Vol. 2, Issue 6, Jun 2013.
[11] Bangyong Sun and Shisheng Zhou, "Study on the 3D Interpolation Models Used in Colour Conversion", International Journal of Engineering and Technology, Vol. 4, No. 1, February 2012.
[12] Scale Space Theory [Online] Available: http://www.cse.iitm.ac.in/~vplab/courses/CV_DIP/PDF/SCALE_SPACE-PYR.pdf.
[13] ADSP-BF533 architecture [Online] Available: http://www.analog.com/en/products/processors-dsp/blackfin/adsp-bf533.html#product-overview
[14] ADSP-BF533 Ez-Kit Lite [Online] Available: http://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/bf533-ezlite.html
[15] Analog Devices VDSP++ Software tool [Online] Available: http://www.analog.com/en/design-center/processors-and-dsp/evaluation-and-development-software/vdsp-bf-sh-ts.html





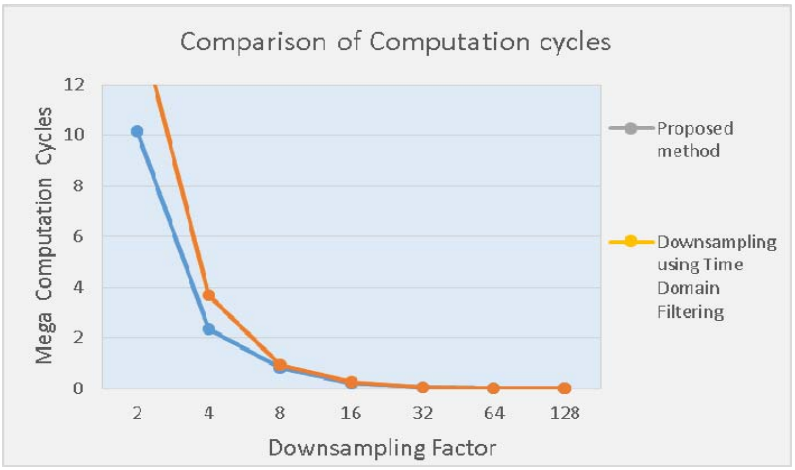


Fig. 6. Comparison of Computations for input images of (top) 1920x1080 resolution, (middle) 800x533 resolution and (bottom) 640x480 resolution

Fig. 7. (A) Original image with resolution 1920 x 1080 (B) Frequency Domain output (C) Time Domain output

Fig. 8. (A) Original image with resolution 800 x 533 (B) Frequency Domain output (C) Time Domain output

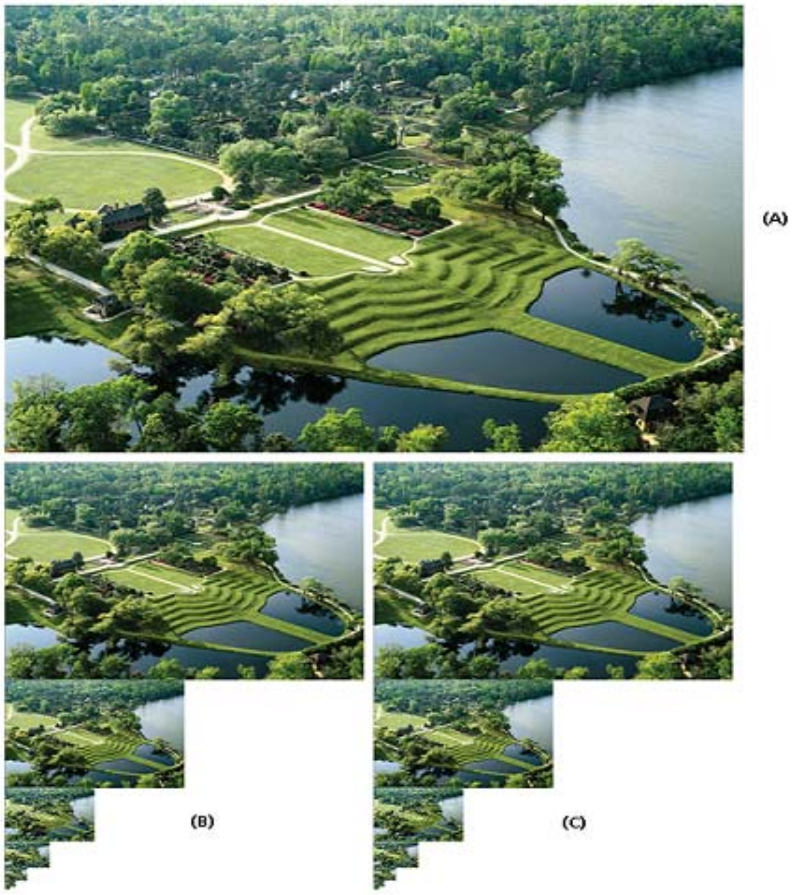


Fig. 9. (A) Original image with resolution 640 x 480 (B) Frequency Domain output (C) Time Domain output