IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 8, August 2015.
www.ijiset.com
ISSN 2348 – 7968

Energy Resilient & Power Aware Approximate Processing Unit for Depth Image Based Rendering Process

Manish Mahant[1], Sapna Choudhary[2]

[1] M.Tech. Scholar (CSE), SRGI,
Jabalpur, M.P., India

[2] HOD (M.Tech, CSE), SRGI,
Jabalpur, M.P., India

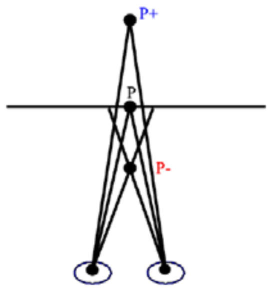
Abstract

In present ere every human demands number of multimedia application. Those applications are based on graphics and computer vision. Similar in this era every one want realistic view so there is tremendous demand of Three- dimension (3D) technology. As we know 3D technology increases the visual quality as compared to two-dimensional (2D) technology. For generation of 3D content there is need of Depth image based rendering (DIBR) process which will generate left and right image through depth image and original image. Basically DIBR is following the concept of actual 3D recording camera setup. Through original camera setup there is virtual camera formula is generated which will create left and right image. Using both image 3D content is created. As we already know for any image processing application time complexity is main issue. So in this work I will propose a fast and approximate DIBR algorithm which will reduce the time complexity issue. In this work we will represent the hardware and algorithm implementation of our proposed design. Here image quality measurement there is some scientific parameter will use which will check the quality of generated left and right image through proposed DIBR algorithm. Those parameters are like PSNR, SSIM, RFSIM and FSIM. Algorithm will implement on Matlab.

Keywords: *FPGA, FSIM, Gaussian, GMSD, PSNR, SSIM, RFSIM.*

1. Introduction

Depth perception in multimedia is something that has seen a great rise in interest in recent years. 3DTVs have made their way into the living rooms of families, 3D cinema releases are more frequent and the quality is leaps and bounds greater than what was seen before. The sudden boom can be seen as a resurrection of a past technology which, due to more recent advancements in both display technologies and content generation, can now be enjoyed by more people and with a higher level of quality. Specially, the type of 3D which is the focus of this thesis and is used in describing today's 3DTVs or movies is known as Stereoscopic 3D and the process through which depth is perceived using stereoscopic images is known as Stereoscopy. Stereoscopic 3D is the illusion of 3D depth created by using a pair of left and right images, henceforth referred to as a stereoscopic pair, and making sure each image is only seen by the left and right eyes respectively. The brain is accustomed to creating depth in the world around us from what is seen with both eyes, because each eye does not see the same thing as the other and so the disparity between the views in each eye is one of the ways the brain recognizes depth. This particular depth cue is referred to as parallax; Figure below illustrates what is meant by parallax. In Figure point P lies on the stereo plane and corresponds to objects which will appear at on a screen when viewed. Negative parallax would be for objects which appear in front of the screen and positive parallax is for objects further away from the screen. By manipulating the separation between pixels in left and right images the different types of parallax can be achieved.[1]



Anaglyph Rendering of an Image from the Superman Cartoon

2. System Analysis

Existing System

Depth-Image Based Rendering (DIBR):

Depth-image-based rendering (DIBR) is the process of synthesizing "virtual" views of a scene from still or moving images and associated per-pixel depth information[2,3]. Conceptually, this novel view generation can be understood as the following two-step process: At first, the original image points are re projected into the 3D world, utilizing the respective depth data. After this, these 3D space points are projected into the image plane of a "virtual" camera, which is located at the required viewing position.

Stereoscopic Image Creation

On a stereoscopic- or auto stereoscopic 3D-TV display, two slightly different perspective views of a 3D scene are reproduced simultaneously on a joint image plane. The horizontal differences between these left- and right-eye views, the so-called *screen parallax* values, are interpreted by the human brain and the two images are fused into a single, three-dimensional percept**.**

2.1. DIBR Algorithm (Old_DIBR)

The process of generating virtual views from mono scope and associated depth is the process of DIBR. Consider an virtual 3D space point M [1], the two side by side projection equations in two views result to:

$$\widetilde{m} = AP_n\widetilde{M}, \widetilde{m}^* = A^*P_nD\widetilde{M}.$$

Where *m* and *m*∗ shows two 2D image points in the left and right view, respectively. The matrix D is of the rotational matrix and the translational matrix *t* that transforms the 3D point from the regular angle system into the camera angle/ coordinate system. Matrices A and *A*∗ specify the essential parameters of cameras. The identity matrix P_n designates the normalized perspective projection matrix. Fig shows a virtual camera setup for altering of virtual angle viewing. The parameters *f* and *tc* represent the focal length and the baseline distance between two virtual angle camera *Cl* and *Cr*, respectively. *Zc* means the depth value of the ZPS (Zero Parallax Setting). The side by side camera setup rather than the convergent camera setup is used not to generate vertical inconsistency and it's much easier to implement with DIBR. Since the position of the 3D space point M depends on its depth value, the 3D image warping equation can be obtained :

$$\widetilde{m}^* = \widetilde{m} + \frac{A^*t}{depth} + \begin{bmatrix} h \\ 0 \\ 0 \end{bmatrix}, with \quad t = \begin{bmatrix} t_x \\ 0 \\ 0 \end{bmatrix}.$$

the pixel position (*x*, *y*) of warped image point can be calculated as:

$$x^* = x + \frac{\alpha_u t_x}{depth} + h, with \quad y^* = y$$

Where *αu* is a parameter which represent the distance of left-right-eye and screen. The pixel position (*xc*, *y*), (*xl*, *y*) and (*xr*, *y*) of the reference view and two virtual views corresponding to the point *P* with *depth* have the following relationship:

$$x_l = x_c + \frac{t_c f}{2depth(x_c, y)} + h,$$
$$x_r = x_c - \frac{t_c f}{2depth(x_c, y)} - h$$

The offset h between reference view and target view can be computed by:

$$h = -\frac{t_c f}{2Z_c}$$

Algorithm Optimization

The next step after hole-filling is the whole left-and-right view images are to be transformed into horizontally, by replacing h pixels for addition and subtraction of h[6]. The division operation is one which consumes time and area by hardware implementation, the DIBR algorithm is optimized to eliminate the division operation. 1*/depth* ranges between 0 and 1. Since the value of 1*/depth* only represents the relative distance of the pixels but not real distance, and that people are only sensitive about the objects which have smaller depth value, (256−*depth*)*/*256 can be used to replace 1*/dept[4]*.

$$x_l = x_c + k\frac{t_c f}{2}\frac{256 - depth(x_c, y)}{256},$$
$$x_r = x_c - k\frac{t_c f}{2}\frac{256 - depth(x_c, y)}{256}$$

In the implementation, for each depth value between 0 and 255, the average drifted between (the practical value) and (the theoretical value) is about 3.6%[1]. The division operation can be simply implemented by shift operation in hardware. In the implementation, the white color in the depth map represents the nearest plane while the black color (i.e. 0) represents the farthest. It is opposite to the real depth given in[5] .We use *D* to represent the value that obtained from the depth map and get:

$$D = 256 - depth$$

L_ of f set and R_ of f set are used to represent the value of off set in the left-and-right view images, respectively. So while implementing the DIBR algorithm in hardware, we use the above equation to compute left-view offset l offset and right-view offset r offset:

$$\begin{cases} l\_offset = D \cdot pos/256 \\ r\_offset = pos - l\_offset. \end{cases}$$

Where *pos* = *k tcf* 2 is a parameter related to the eye-screen distance and the screen size. *pos* is set to 1/32 of the width of the screen[6]. This give real view and can simply implemented by shifting operation. After the *l offset* and *r offset* are calculated, the left-view image and the right-view image can be obtained.

$$\begin{cases} lpic(x,y) = pic(x - l\_offset, y) & left - view \\ rpic(x,y) = pic(x - r\_offset, y) & right - view. \end{cases}$$

Where *lpic* and *rpic* represent the left-view and rightview images, respectively. *pic* is the original 2D image, and (*x*, *y*) means the pixel position in the images.
Pre-Processing of depth image is usually a smoothing filter. Because depth image with horizontal sharp transition would result in big holes after warping, smoothing filter is applied to smooth sharp transition so as to reduce the number of big hole. However, if we blur the whole depth image, we will not only reduce big holes but also degrade the warped view. This is because the depth map of non-hole area is smoothed.[7]

2.2. New DIBR Algorithm

In this algorithm author proposed a new approach. According to this approach he will try to reduce the hole filling problem. In this algorithm author proposed a new formula for generation of Left and Right Image. According to this formula they are generating left and right image in which they are fill hole by using of corresponding depth map. Problem with approach is that it will require heavy amount of time means this algorithm will increase time complexity.

$$X_{\text{view}} = X_C + (n + \delta - i)(-1)^{\alpha} \frac{t_x f}{n-1} \cdot [(\frac{1}{v_f})(\frac{v}{v_f}) + (1 - \frac{v}{v_f})]$$

$$\alpha = \begin{cases} 1, & \text{if } X_{\text{view}} = X_l \\ 0, & \text{if } X_{\text{view}} = X_r \end{cases}$$

$$\delta = \begin{cases} 1, & \text{if } n = odd\ integer \\ \frac{1}{2}, & \text{if } n = even\ integer \end{cases}, \text{ when } Z < Z_{fp}.$$

3D Image Warping

3D image warping maps intermediate view pixel by pixel to left or right view according to pixel depth value. In the other words, 3D image warping transforms pixel location according to depth value. The 3D image warping formula is as following:

$$x_l = x_c + (\frac{t_x}{2} \frac{f}{Z}),$$

$$x_r = x_c - (\frac{t_x}{2} \frac{f}{Z})$$

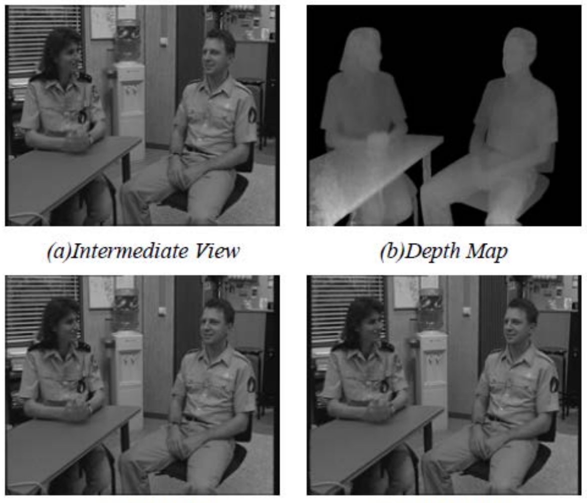
The x*l* is the horizontal coordinate of the left view, and *xr* is the horizontal coordinate of the right view. Besides, *xc* is the horizontal coordinate of the intermediate view. *Z* is depth value of current pixel, *f* is camera focal length and t*t x* is eye distance. The formula shows that 3D warping maps pixel of intermediate view to left and right view in horizontal direction.

Hole Filling

Average filter interpolation method is a common method for Hole-Filling in DIBR, which only would result in artifacts at highly textured areas. Apart from this the hole-size in DIBR is so big that it is needed to using average filter with large window size. At the same time, average filter with large window size cannot store edge information for the reason that edge information is blurred. The formula used to do Hole-Filling is as below:

$$\frac{\sum_{v=-w}^{v=w} \left\{ \sum_{\mu=-w}^{w} s(x-\mu, y-v) \times non\_hole(x-\mu, y-v) \right\}}{\sum_{v=-w}^{v=w} \left\{ \sum_{\mu=-w}^{w} non\_hole(x-\mu, y-v) \right\}}$$

$$non\_hole(x,y) = \begin{cases} 0 & \text{If } (x, y) \text{ is a hole} \\ 1 & \text{Otherwise} \end{cases}$$


(a)Intermediate View (b)Depth Map

(c) Left View (d) Right View

2.3.RGB to YCbCr: (Color space transformation) (Latest_DIBR)

The image in RGB format should be converted into different luminance based color formats called as YCbCr . There are three components in Ycbcr :

1. Y component represents the brightness of a pixel
2. Cb and Cr components represent the chrominance (split into blue and red components).

The same YCbCr used in digital color television, Video DVD's and even in analog it is similar way color is represented in analog PAL video and MAC, where NTSC uses the YIQ color space. The YCbCr color space conversion allows a high rate of compression without a significant effect on perceptual image quality (or greater perceptual image quality for the same compression). The compression is more efficient in terms of brightness information, which is more important to the eventual perceptual quality of the image, is limited to a single channel, which more close to the human visual system. This conversion to YCbCr is specified in the JFIF standard, and should be performed for the resulting JPEG file to have maximum compatibility. Apart from some JPEG implementations which does not use this type and which store in same channel of RGB. This type results in less efficient compression and can't able to use for larger size of images.

ITU-R BT.601 conversion [5]

The representation of Y′CbCr is as shown below which was defined for standard-definition television use by the ITU-R BT.601 (formerly CCIR 601) standard . The use with digital component video is derived from the corresponding RGB space as follows:

$$K_B = 0.114$$
$$K_R = 0.299$$

From the above constants and formulas, the following can be derived
Analog YPbPr from analog R'G'B' is derived as follows:

$$\begin{aligned} Y' &= \quad 0.299 \cdot R' + 0.587 \cdot G' + 0.114 \cdot B' \\ P_B &= -0.168736 \cdot R' - 0.331264 \cdot G' + 0.5 \cdot B' \\ P_R &= \quad 0.5 \cdot R' - 0.418688 \cdot G' - 0.081312 \cdot B' \end{aligned}$$

Digital Y′CbCr (8 bits per sample) is derived from analog R'G'B' as follows:

$$\begin{aligned} Y' &= 16 + (65.481 \cdot R' + 128.553 \cdot G' + 24.966 \cdot B') \\ C_B &= 128 + (-37.797 \cdot R' - 74.203 \cdot G' + 112.0 \cdot B') \\ C_R &= 128 + (112.0 \cdot R' - 93.786 \cdot G' - 18.214 \cdot B') \end{aligned}$$

$$(Y', C_B, C_R) = (16, 128, 128) + (219 \cdot Y, 224 \cdot P_B, 224 \cdot P_R)$$

The range of resultant signals is from 16 to 235 for Y' (Cb and Cr range from 16 to 240); the values which lies in range of 0 to 15 are called *foot room*, while the values in range of 236 to 255 are called *headroom*.
Another possibility is, digital Y′CbCr can derived from digital R'dG'dB'd (8 bits per sample, each using the full range with zero representing black and 255 representing white) as per the following equations:

$$\begin{aligned} Y' &= 16 + \frac{65.738 \cdot R'_D}{256} + \frac{129.057 \cdot G'_D}{256} + \frac{25.064 \cdot B'_D}{256} \\ C_B &= 128 - \frac{37.945 \cdot R'_D}{256} - \frac{74.494 \cdot G'_D}{256} + \frac{112.439 \cdot B'_D}{256} \\ C_R &= 128 + \frac{112.439 \cdot R'_D}{256} - \frac{94.154 \cdot G'_D}{256} - \frac{18.285 \cdot B'_D}{256} \end{aligned}$$

In the above formula, the scaling factors are multiplied by 256/255. This allows for the value 256 in the denominator, which can be calculated by a single bit shift. If the

R'dG'dB'd digital source includes foot room and headroom, the foot room offset 16 needs to be subtracted first from each signal, and a scale factor of 255/219 needs to be included in the equations.

The inverse transform is:

$$R'_D = \frac{298.082 \cdot Y'}{256} + \frac{408.583 \cdot C_R}{256} - 222.921$$
$$G'_D = \frac{298.082 \cdot Y'}{256} - \frac{100.291 \cdot C_B}{256} - \frac{208.120 \cdot C_R}{256} + 135.576$$
$$B'_D = \frac{298.082 \cdot Y'}{256} + \frac{516.412 \cdot C_B}{256} - 276.836$$

The inverse transform without any rounding's (using values coming as per ITU-R BT.601 recommendation) is:

$$R'_D = \frac{255}{219} \cdot (Y' - 16) + \frac{255}{112} \cdot 0.701 \cdot (C_R - 128)$$
$$G'_D = \frac{255}{219} \cdot (Y' - 16) - \frac{255}{112} \cdot 0.886 \cdot \frac{0.114}{0.587} \cdot (C_B - 128) - \frac{255}{112} \cdot 0.701 \cdot \frac{0.299}{0.587} \cdot (C_R - 128)$$
$$B'_D = \frac{255}{219} \cdot (Y' - 16) + \frac{255}{112} \cdot 0.886 \cdot (C_B - 128)$$

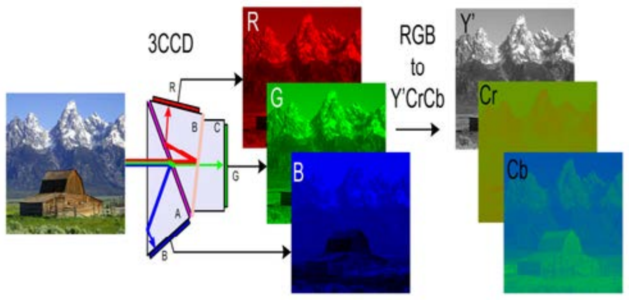
This form of Y′CbCr is used primarily for older standard-definition television systems, as it uses an RGB model that fits the phosphor emission characteristics of older CRTs.

$$Y = \frac{77}{256} E'_{R_D} + \frac{150}{256} E'_{G_D} + \frac{29}{256} E'_{B_D}$$

$$C_R = \frac{131}{256} E'_{R_D} - \frac{110}{256} E'_{G_D} - \frac{21}{256} E'_{B_D} + 128$$

$$C_B = -\frac{44}{256} E'_{R_D} - \frac{87}{256} E'_{G_D} + \frac{131}{256} E'_{B_D} + 128$$

Then the nearest integer coefficients ,base is considered as 256. To obtain the 4:2:2 components *Y*, *CR*, *CB*, low-pass filtering and sub sampling must be performed on the 4:4:4 *CR*, *CB* signals described above. Note should be taken that slight differences could exist between *CR*, *CB* components derived in this way and those derived by analogue filtering prior to sampling.



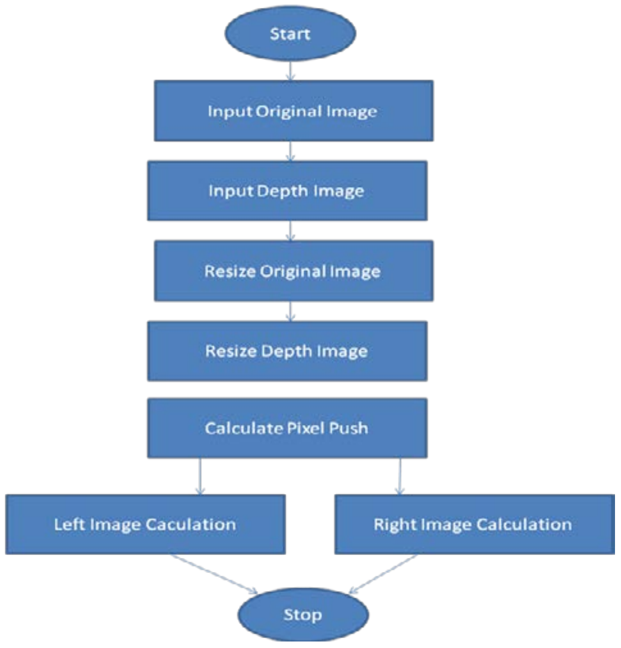
Parameters to evaluate the Image:

Various parameters are used to evaluate the proposed algorithm at both levels. The various parameters are
1. PSNR
2. SSIM
3. FSIM
4. (GMSD)
5. (RFSIM)

Research Gap

The above algorithm of DIBR has the following problem:
a. Previous DIBR model having the problem of timing complexity.
b. All previous approach are not make justice with SPAA (Speed, power, area and Accuracy)

2.4. Proposed DIBR



In this proposed Depth Image based rendering approach we are calculating *Pixel_Shift*. Here *Pixel_Shift* is use for shift the original pixel value with its depth valure. Aftr the calculation of shift value we are apply that calculated *Pixel_Shift* in left and right image generation unit. Using this shift value we will calculate let image and right image. Using this approach when we are implementing hardware unit so there no need of any extra hardware unit which is require in previous algorithm

$$Pixel\_Shift = Displacment * ([Depth]/255$$

General formula of Displacement is (*k* tc*f)/* 2 is the value of the parameter related to the distance of eye and the distance between screen. But here Displacement is set to 1/32 of the width of the screen . Zc is convergence Distance. This will provide a good visual experience and can also calculated simply by shifting operation. After the *Pixel_shift value calculation* , the left-view image and the right-view image can be obtained.
In the above formula previously left_push and right_push are calculated individually, where in our approach only one calculation is required for calculating.

$$Xle(x,y) = pic(Xc - Pixel\_Shift)$$

$$Xri(xy) = pic(Xc + Pixel\_Shift)$$

Where *Xle* and *Xri* is used to represent images as left-view and rightview, respectively. *pic* is the original 2D image, and (*x, y*) means the pixel position in the images. While caluculating Xre pic we add Pixel_Shift and for calculation Xl we substrate Pixel_Shift.

3. Results and Analysis

In this section comparative analysis is perform for different existing and proposed RGB to YCbCr and Depth image based rendering process. Analysis is performing by using of MATLAB. Here generated image from different approach is pass from image quality measurement parameter which are:

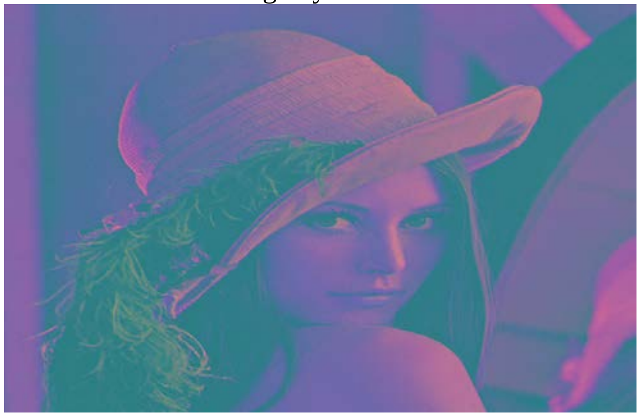
- PSNR
- SSIM [14]
- RFSIM [12]
- FSIM [13]
- Absolute Similarity (%)

RGB to YCbCr Conversion:

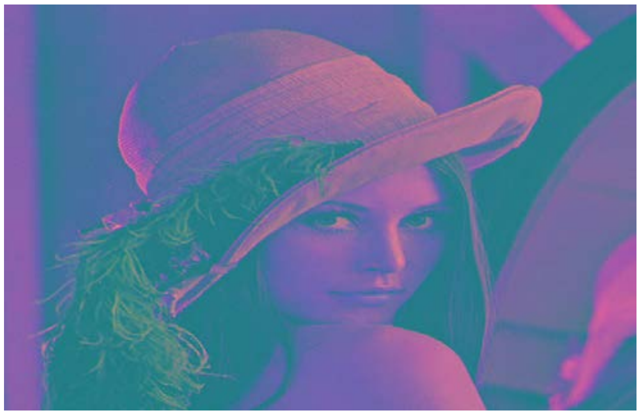
Using of these parameter image qualities is measured. At initial stage RGB to YCbCr analysis is perform. ITU proposed RGB to YCbCr conversion unit which is compare with proposed RGB to YCbCr unit. Here standard lena image(512 X 512) is used for analysis point of view.

Original Input Image:

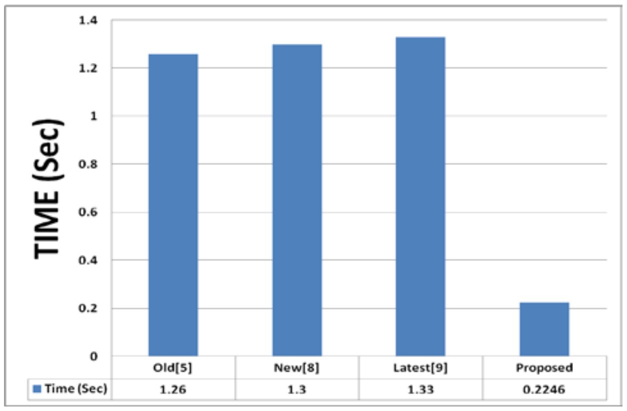
Standard YCbCR Image by Matlab:



YCbCr Image by Proposed Approach:



Time Complexity Comparative Results:



As we can see in this result proposed DIBR approach is better than existing approaches like DIBR OLD and DIBR MODIFIED. Proposed approach requires approximate 40% reduction in time complexity as compare to DIBR OLD & MODIFIED.

Image Quality Parameter Results:

Parameter	Old	New	Latest	Proposed
PSNR	16.35	18.755	22.45	22.31
SSIM	0.8377	0.86000	0.9310	0.8808
RFSIM	0.1195	0.3202	0.4277	0.2214
FSIM	0.9062	0.9454	0.9580	0.9272
GMSD	0.8515	0.8998	0.8900	0.8990
Similarity	72.12	75.84	90.11	85.01

4. Conclusions

In this work we represent the error resilient and power aware architecture and algorithm of depth image based rendering approach. In this work we proposed an architecture which require less power and area. Simmilar proposed approach require very less delay time at algorithm so complexity level in this proposed approach is very less as compare to previous exsisting approach of [5],[8] and [9]. Here as wecan see hardware complexity is reduces by 50% as compare to DIBR latest [9]. In proposed approach there is very small deridation in image quality which is tolerable by human eye. In this approach we are proposed a new algorithm for RGB to YCbCr conversion and DIBR. In proposed approach time complexity is reduce and generated output image quality is measured by using of some image quality parameter like PSNR, SSIM, FSIM, RFSIM, GMSD, Similarity (%). Here simulation result shows time complexity is reduces by 80% and hardware complexity is reduces by 30 to 40%.

References

[1] C. Fehn, “A 3D-TV Approach Using Depth-Image-Based Rendering (DIBR)”, proceeding of *SPIE* 5291, Stereoscopic Displays and Virtual Reality Systems XI, 93 (May 21, 2004).
[2] Kwanghee Jung, Young Kyung Park, Joong Kyu Kim, Hyun Lee, Kugjin Yun, Jinwoong Kim, “Depth Image Based Rendering for 3D Data Service Over T-DMB,” The True Vision-Capture, Transmission and Display of 3D video, 2008, vol. no. , pp. 237-240, 28-30 may 2008.
[3] A Hang Shao; Xun Cao; Guihua Er, " Objective quality assessment of depth image based rendering in 3DTV system ", DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video , 2009 , vol., no., pp.1,4, 4-6 May 2009.
[4] Nguyen, Q.H.; Do, M.N.; Patel, S.J.,, " Depth image-based rendering with low resolution depth ", Image Processing (ICIP), 2009 16th IEEE International Conference on, vol., no., pp.553,556, 7-10 Nov. 2009.
[5] Rec. ITU-R BT.601-5, Studio Encoding Parameters of Digital Television for Standard 4:3 and Wide-screen 16:9 Aspect Ratios, (1982-1986-1990-1992-1994-1995), Section 3.5.
[6] Cheng Fan; Yi-Chun Chen; Shih-Ying Chou,” Vivid-DIBR Based 2D–3D Image Conversion System for 3D Display,” Display Technology, Journal of , vol.10, no.10, pp.887,898, Oct. 2014doi: 10.1109/JDT.2014.2331064
[7] Chao-Chung Cheng; Chung-Te Li; Liang-Gee Chen, " A novel 2Dd-to-3D conversion system using edge information ", Consumer Electronics, IEEE Transactions on, vol.56, no.3, pp.1739,1745, Aug. 2010.