High-Quality View Interpolation Based on Depth Maps and Its Hardware Implementation

Yanzhe Li, Kai Huang
Institute of VLSI Design
Zhejiang University
Hangzhou, China
Email: {liyz,huangk}@vlsi.zju.edu.cn

Luc Claesen
Engineering Technology - Electronics-ICT Dept.
Hasselt University
3590 Diepenbeek, Belgium
Email: luc.claesen@uhasselt.be

***Abstract*—Three dimensional (3D) vision applications have drawn more attention nowadays and many products are entering the mass market. View interpolation is a crucial step to generate intermediate viewpoints from reference images. However, it is still challenging to achieve good performance in both processing speed and image quality for various 3D applications. In this paper, a hardware-compatible view interpolation algorithm is proposed, which can produce high-quality intermediate images by disparity warping and color blending. Moreover, a fully pipelined hardware architecture is designed based on the algorithm. A prototype of the proposed architecture has been implemented on an Altera Stratix-IV FPGA board, achieving 65 frames per second (fps) with a full HD (1920 $\times$ 1080) resolution. It is evaluated on the Middlebury benchmark quantitatively, and visual results of real-world images are also provided.**

I. Introduction

Three dimensional (3D) vision applications, such as three dimensional television (3D-TV) and virtual reality gaming, are rapidly growing in popularity as many products are currently entering the mass market. They provide the audience with a greater sense of presence in a computer-generated environment. Specific viewing technology is used to ensure that two separate streams of images are provided for two eyes from different viewpoints. One stream consisting of images captured with a camera viewpoint that is intended for the left eye and the other stream with a viewpoint intended for the right eye. However, the requirement to wear additional eyeglasses is usually perceived uncomfortable. Recently, autostereoscopic displays (we call them 3D displays) support glasses-free 3D depth perception by showing multiple views simultaneously so that the viewer always sees a stereo pair from predefined viewpoints regardless of his/her position [1]. The large number of consecutive views, which are arranged properly to alleviate the motion parallax conflict, has significantly increased the amount of data to be processed and transmitted. Consequently, the need to render additional virtual views from transmitted views arises, in order to support 3D displays.

The basic principle to generate new views is based on projective transformation [2]. Given the depth information of each pixel in an image and the camera matrices consisting of extrinsic/intrinsic camera parameters, pixels in the reference image can be deprojected from screen coordinates into world coordinates and eventually projected into the virtual camera view. As a result, a 3 $\times$ 3 homography matrix is used to describe the relationship between two corresponding pixels. For a horizontally rectified camera configuration, the projection is simplified to the horizontal direction, i.e. corresponding pixels must lie in the same row of the images. In this vein, an algorithm of view interpolation to efficiently generate intermediate viewpoints is required.

View interpolation, which is treated as the key role in a 3D display system, takes two rectified stereo images as input and generates intermediate viewpoints based on their disparity maps and intermediate position. Note that it is a complicated and time-consuming procedure, but 3D applications would require high-quality performance and real-time processing speed. As a result, it is difficult for software implementations on a CPU to meet these constraints. In this condition, hardware acceleration is inevitable and it has been done extensively using GPUs and dedicated hardware.

The techniques to generate virtual views from a number of real images have been studied since the 1990s. Traditional methods synthesize novel views from the color-textured 3D models but suffer from the complex and unreliable process. A number of approaches have been proposed to address this problem. Image morphing is introduced as a fast method to generate new view images without an explicit 3D model [3]. An adaptive technique based on the block-wise disparity estimation is proposed to interpolate intermediate views in [4]. The left and right images are both projected and then the intermediate view is created using a weighted average of two projected images. Nowadays, the MPEG reference software for view synthesis (VSRS) [5] is widely used; it can be updated with improvements through the MPEG standardization process. Several sub-algorithms, such as half-pixel precision and temporal enhancement, have been integrated in the VSRS to achieve high-quality results.

The prior works mentioned above always focus on the image quality of the intermediate views. However, their complicated algorithms make them unviable for real-time applications. A performance-efficient architecture of the view interpolation algorithm is mapped onto a synchronized pair of Virtex-5 FPGA boards in [6], but the quality of the intermediate views is not illustrated. A free viewpoint rendering algorithm is implemented on GPUs as a real-time implementation for high

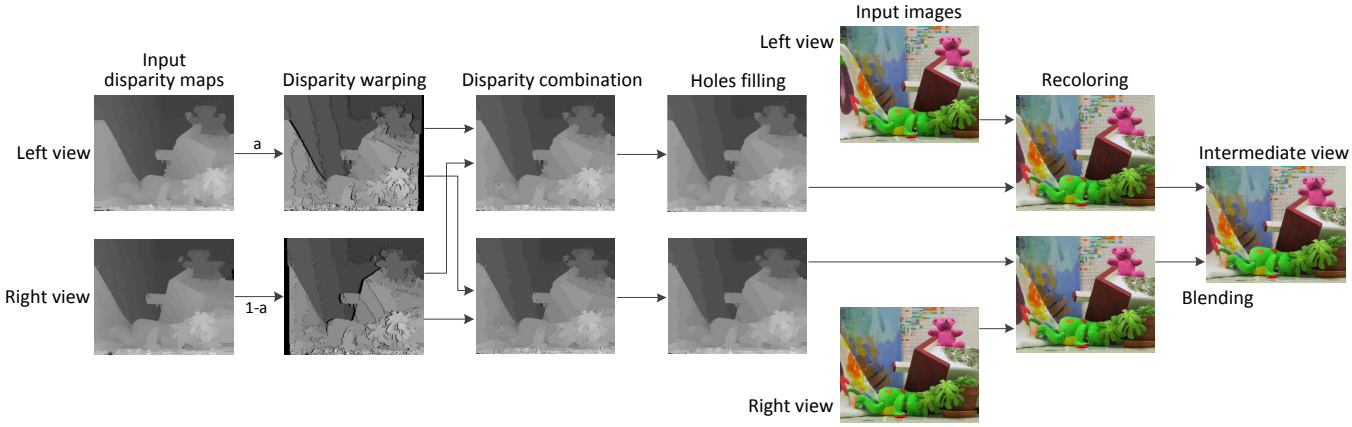

Fig. 1. The whole procedures of the EDWCB algorithm.

resolution multi-view videos [7]. Although GPU has proved to be an attractive speedup platform for rendering algorithms, high power consumption restricts its performance. A real-time high-resolution free viewpoint synthesis hardware system utilizing three-camera disparity estimation is presented in [8]. However, only the real-world images of the intermediate views are displayed but no quantitative quality results are provided to make it convincing.

In this paper, an enhanced disparity-warping color blending (EDWCB) algorithm is proposed to generate novel intermediate views with two reference images and their corresponding disparity maps. Different from conventional methods that synthesize the color image immediately, an intermediate view is generated by warping the disparity maps and then merging each other to fill in new exposed areas. Further, the warped disparity maps are enhanced in a more consistent manner before recoloring the final image. On the basis of the proposed algorithm, a fully pipelined architecture is presented. A prototype of the hardware system is implemented on an Altera FPGA, achieving 65 frames per second (fps) for full HD (1920 $\times$ 1080) resolution. The design is evaluated on the Middlebury benchmark quantitatively, and visual results of real-world images are also provided.

In the rest of this paper, Section II presents the proposed view interpolation algorithm. The hardware implementation based on the algorithm is described in Section III. Experimental results are shown in Section IV and the paper is concluded in Section V.

II. View Interpolation Algorithm

A. Algorithm Overview

In the proposed view interpolation algorithm EDWCB, a novel intermediate viewpoint is generated and can be placed anywhere on the horizontal baseline between two reference images. It takes two rectified stereo images as input and warps them to the intermediate view, based on their disparity maps and the intermediate position. The EDWCB algorithm consists of five steps: disparity warping, disparity combination, holes filling, image recoloring, and image blending. First, the input disparity maps are warped to the desired viewpoint. Then the warped disparity maps are combined to each other for occluded regions, after which small holes are filled by their neighboring information. These two steps can be treated as an enhanced refinement to the disparity warping step. Finally, two images are recolored simultaneously and blended together to produce the final synthesized color image. The whole procedures of the EDWCB algorithm are illustrated in Fig. 1 and the Middlebury *Teddy* images [9] are used to show the visual effect of each step.

B. Disparity warping

To generate the intermediate view, the disparity maps are warped to the desired intermediate position by the parameter $a \in [0, 1]$, where $a = 0$ represents the left viewpoint and $a = 1$ represents the right viewpoint. The warped left disparity map d_{wl} is defined as

$$d_{wl}(x_p, y_p) = a \times d_l(x_p + a \times d_l(x_p, y_p), y_p) \qquad (1)$$

where d_l is the left input disparity map and (x_p, y_p) represents pixel coordinate in the warped image. Similarly, the warped right disparity map d_{wr} is defined as

$$d_{wr}(x_p, y_p) = a' \times d_r(x_p - a' \times d_r(x_p, y_p), y_p) \qquad (2)$$

where the warping direction is reversed and the parameter is changed to a'. Since both disparity maps are warped to the same intermediate position, a' is defined as: $a' = 1 - a$.

Note that the disparity warping is not a one-to-one mapping; there may be two or more disparities warped to the same location in the intermediate view. In this case, the largest disparity is always retained, while smaller disparities are overwritten. In the case of the parallel stereo camera configuration, foreground objects that have large disparities may cover background objects with small disparities. Thus, not all locations in the intermediate image can be filled in; the warped disparity maps still contain invalid edges and gaps, shown as pure black patches in the disparity warping step in Fig. 1. These patches are mainly caused by occlusions as well as caused by mismatches in the input disparity maps. If the

parameter a is closer to 0, the patches in d_{wl} will be less severe than those in d_{wr}. Otherwise, the condition is inverse. In Fig. 1, a is defined as 0.5, which means that the desired viewpoint is just in the middle of the two reference images.

C. Disparity combination and holes filling

To handle these patches, the two warped disparity maps, d_{wl} and d_{wr}, must be combined with each other. For d_{wl}, the pixels in the occluded regions have no correspondence in the left image and thus need to be determined based on the right image. In this condition, d_{wr} is used to fill in the patches in d_{wl}. The combined left disparity map d_{cl} is expressed as

$$d_{cl} = \begin{cases} d_{wl} & \text{if } d_{wl}(x_p, y_p) = VALID \\ d_{wr} & \text{else if } d_{wr}(x_p, y_p) = VALID \\ INVALID & \text{else} \end{cases}. \quad (3)$$

The combined right disparity map d_{cr} can also be calculated in the same way.

As shown in the disparity combination step in Fig. 1, most patches are filled in; meanwhile a few disparities are still invalid, which can be treated as small holes. The procedure of filling in these small holes is to update the invalid disparity with the most frequent valid disparity in a 5×5 support window first. Then any remaining invalid disparities are filled in using the closest valid disparity in scanline order. Special care should be taken to differentiate disparities warped from the left and the right viewpoints. At the end, the final disparity maps of the intermediate viewpoint are shown in the holes filling step of Fig. 1, where no invalid disparities are remaining.

D. Image recoloring and blending

In the image recoloring step, the refined disparity maps d_{cl} and d_{cr} are recolored to produce two synthesized images I_{la} and I_{ra}, respectively. If a pixel of the intermediate image contains a disparity coming from d_{wl}, its associated color must be fetched from the left image I_l. Otherwise, its color would be fetched from the right image I_r. The projected image I_{la} (I_{ra}) can be expressed as

$$I_{la(ra)} = \begin{cases} I_l(x_p + d_{cl(cr)}, y_p) & \text{if } d_{cl(cr)} = d_{wl} \\ I_r(x_p - d_{cl(cr)}, y_p) & \text{else} \end{cases}. \quad (4)$$

If the calculated pixel location (x and y coordinate) is not an integer in the reference images, the nearest pixel with the integer x and y coordinates in the same line will be used to avoid complex operations.

The projected images are usually different from each other because I_{la} is derived from I_l while I_{ra} is derived from I_r. Additionally, the occluded regions of I_{la} are also different from those of I_{ra}. Therefore, the final intermediate view I_a can be generated by blending the two projected images, which is defined as

$$I_a = a' \times I_{la} + a \times I_{ra} \quad (5)$$

where the ratio of the weight coefficients for the projected images should be the inverse of the distance ratio between the viewpoints [10]. Finally, the intermediate view I_a of the *Teddy* stereo images is shown in Fig. 1.

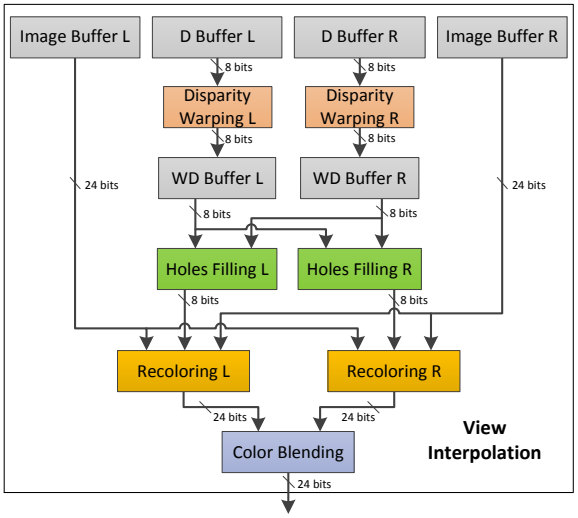

Fig. 2. Block diagram of the proposed architecture.

III. Hardware Implementation

In this section, a hardware architecture is designed based on the proposed algorithm, as shown in Fig. 2. The key to improve its processing ability is to be a fully pipelined architecture without external memory limitation. For both images, source image pixels and disparity maps are stored in buffers as input. The image of the intermediate view is generated in scanline order after a certain pipeline latency. No data is read from the external memory during the whole processing flow, so that memory bandwidth is not a limitation.

In Fig. 2, the 8-bit disparity maps and the 24-bit RGB pixels of both images have been buffered and synchronized. The intermediate position a, which is defined as a 4-bit unsigned integer, is treated as input to this stage. The architecture to generate the intermediate image from the left view is shown in Fig. 3; it is also duplicated to generate the intermediate image from the right view. The *Disparity Warping State-machine* will start once the left disparity buffers are available. The read address is generated to read back the left disparities d_l row by row. Then, the warped disparity values d_{wl}, which are calculated using the left disparities d_l and the intermediate position a, are written into the warped disparity buffers; they are also used to calculate the write address to the warped disparity buffers. In addition, validity arrays, each of which have the size of 1920, are utilized to store the corresponding validity bits. Whenever a disparity is written into the warped disparity buffers, the corresponding address in its validity array is marked as valid. The disparities that are not validated remain as invalid disparities for disparity combination and holes filling.

The *Holes Filling State-machine* works when the warped disparity buffers and the validity arrays are available. The warped disparities d_{wl} and validity bits are read back from the warped disparity buffers and the validity arrays, respectively. The validity bits are used as the selection signal to process the disparity combination. If a warped disparity is valid, it will be

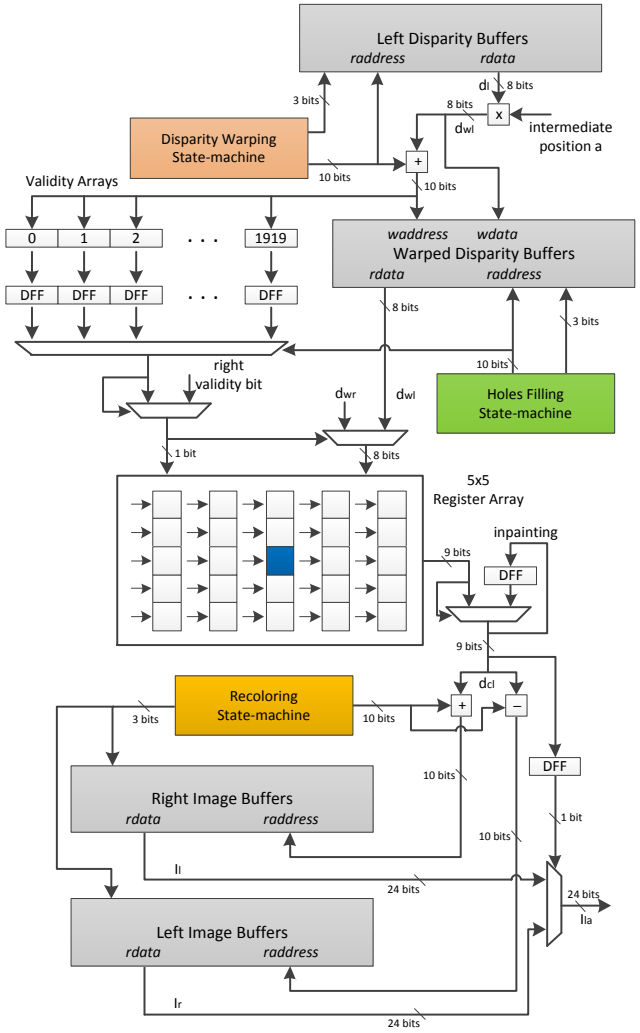

Fig. 3. Architecture to generate the intermediate image from the left view.

shifted into a 5×5 register array together with its validity bit. If a warped disparity is invalid, the disparity value d_{wr} in the same address from the right buffers will be picked up into the array, and its validity bit will be updated. The 5×5 register array is built as the support window for the enhanced refinement, and the center disparity is updated based on the most frequent valid disparity in the window. It is noted that the warped disparities from the left and the right views need to be distinguished. After the voting, if the disparity is still invalid, it will be replaced with the disparity previously stored in the flip-flop.

Then the refined disparity maps d_{cl} are used to calculate the read address to the image buffers. The selection of the image buffers is handled by the *Recoloring State-machine*. If the refined disparities come from the left disparity buffers, the corresponding pixels in the left image I_l will be used. If not, the corresponding pixels in the other image I_r will be selected. In this way, the projected image I_{la} from the left view is derived. On the other hand, the other projected image I_{ra} generated from the right view can be produced in the same way. Finally, the two projected images are blended to generate the desired intermediate image I_a.

IV. Experimental Results

A prototype of the proposed system has been implemented on an Altera EP4SGX230 FPGA board. It is evaluated using rectified synthetic stereo images, which are initially stored in the buffers. In addition, their disparity maps are generated with the stereo matching algorithm proposed in [11] and then synchronized with the stereo images. They are all treated as the input of this system. Resource utilization and image quality of the intermediate viewpoint are important criteria when evaluating the proposed FPGA-based system. These two important aspects are elaborated in the following subsections.

A. Resource Utilization

Table I gives the overall resource utilization of the FPGA prototype. The hardware system utilizes 10% of the ALUTs, 7% of the registers, and 8% of the memory bits on the FPGA board, and can operate at 140 MHz. A frame rate of 65 fps for full HD resolution is achieved in the system. It is noted that only a few hardware resources are utilized on the FPGA board, and the high processing speed of this hardware system can provide a real-time performance. The memory bits are mostly used as buffers for source images and disparity maps to make sure that external memory bandwidth is not a limitation.

TABLE I
Resource Utilization Report

Altera EP4SGX230	ALUTs Total: 228000	Registers Total: 228000	Memory Bits Total: 17133000
View interpolation	22834	15918	1428480

B. Quality Evaluation

To discuss the quality of the proposed system, the intermediate views are evaluated on the Middlebury benchmark using different metrics. For intermediate views, the ultimate validation is whether the views look real or not, but it is also common to use quantitative metrics for quality evaluation, such as peak signal to noise ratio (PSNR) and structural similarity (SSIM) [12].

In the proposed system, intermediate views are produced from a pair of reference images, their disparity maps and the position parameter. Then the virtual intermediate images and the actual images are evaluated quantitatively by both PSNR and SSIM. Here the SSIM is introduced to correlate better with the human visual system, particularly with the artifacts existing in the view interpolation process. Errors such as distortions and ghosts could possibly produce a high PSNR value, since they may occur only in some small parts of the image, whereas they are expected to present a larger error by SSIM. The experiments are run on the 2006 Middlebury benchmark, and the *View1* and *View5* images in the data sets are used as the left and right reference images. In this way, three intermediate images of the quartering viewpoints can be evaluated with the

TABLE II
PSNR AND SSIM COMPARISON OF INTERMEDIATE IMAGES

Data Set	*Aloe*		*Baby1*		*Bowling2*		*Cloth2*		*Flowerpots*		*Rock2*	
Metric	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Previous Methods [13]	25.37	0.862	27.06	0.895	26.91	0.909	25.10	0.863	27.16	0.932	28.42	0.883
Proposed Method	28.17	0.897	34.62	0.954	31.12	0.943	31.33	0.942	28.37	0.948	33.95	0.966

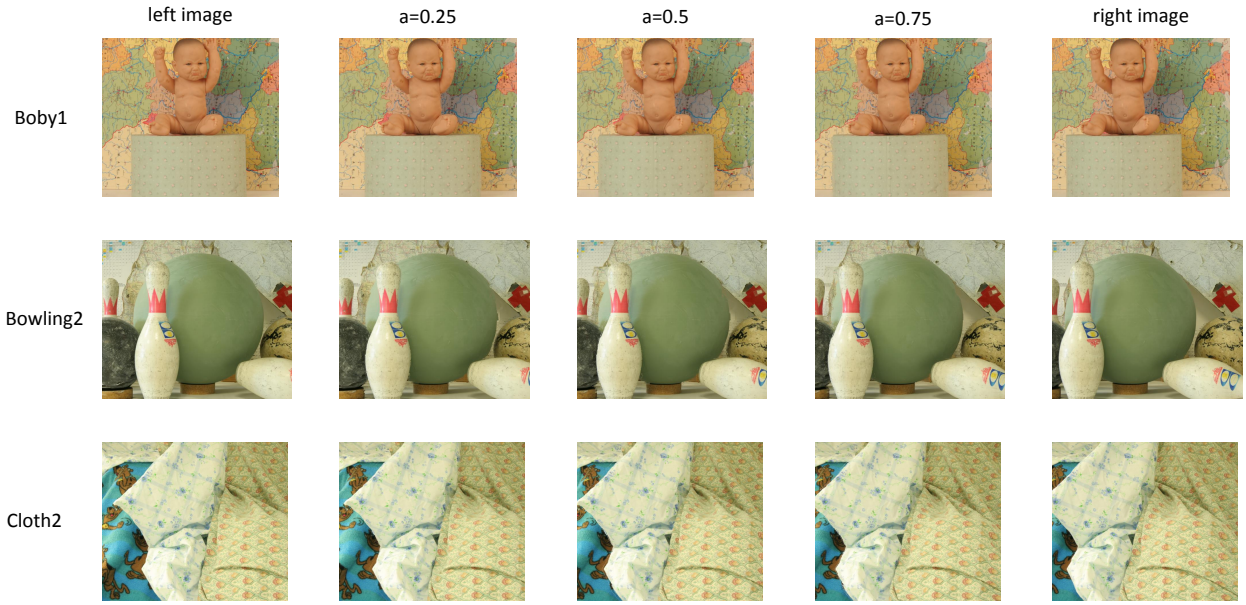

Fig. 4. Visual results of the data sets in the Middlebury benchmark.

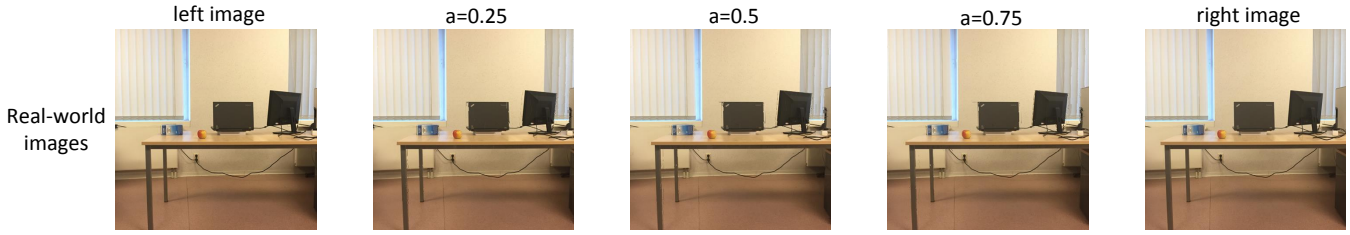

Fig. 5. Visual results of real-world images.

View2, *View3* and *View4* images of the data sets respectively. To make a comparison with previous methods, Table II shows the PSNR and SSIM values at the central viewpoint. In [13], the VSRS produces the virtual views exactly at the center using the disparity maps generated from several different stereo matching algorithms. The intermediate images that have the highest quality in [13] are listed here. It can be clearly noticed that the proposed design outperforms the previous methods in terms of PSNR and SSIM values for all the data sets. The high quantitative results mainly benefit from our EDWCB algorithm.

The visual quality is always a crucial point for the intermediate images. Three intermediate views synthesized for the quartering positions between the two reference images are shown in Fig. 4. Impressive visual results are provided and very few artifacts are observed in the intermediate images. With the help of the enhanced refinement in the view interpolation algorithm, no black patches are remaining, but the most noticeable artifact is the contour of visual disturbances closely surrounding the foreground objects. This halo-like effect is caused by incorrect disparity maps in depth discontinuity regions. Although it is often challenging to improve the accuracy of disparity maps, the visual results show that the proposed system is able to handle this problem adequately.

The reference images in the benchmark are all well captured and rectified so that the results are quite accurate. But the quality of the intermediate views for real-world images may decrease due to some undesirable factors, such as luminance differences and rectification errors. The proposed system is further evaluated by real-world images to prove its robustness, as shown in Fig. 5. The images are captured in the office, and then rectified by software. The visual results show that our

system still provides high-quality intermediate views for real-world images.

V. Conclusion

In this paper, a view interpolation algorithm called EDWCB has been proposed to generate novel intermediate views using reference images and their corresponding disparity maps. In order to implement this algorithm, a fully pipelined hardware architecture is designed. A prototype of the hardware system has been built on an Altera Stratix-IV FPGA, achieving 65 fps for full HD resolution. The design is evaluated on the Middlebury benchmark and visual satisfactory results are provided. In the future, a sub-pixel interpolation method can be presented in the algorithm to improve the quality of intermediate views.

References

[1] N. A. Dodgson, "Autostereoscopic 3D displays," *Computer*, vol. 38, no. 8, pp. 31–36, Aug 2005.

[2] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer, 2010.

[3] S. E. Chen and L. Williams, "View interpolation for image synthesis," in *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '93. New York, NY, USA: ACM, 1993, pp. 279–288.

[4] L. Zhang, D. Wang, and A. Vincent, "Adaptive reconstruction of intermediate views from stereoscopic images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 1, pp. 102–113, Jan 2006.

[5] K. Wegner, O. Stankiewicz, M. Tanimoto, and M. Domanski, "Enhanced view synthesis reference software (VSRS) for freeviewpoint television," *Technical Report ISO/IEC JTC1/SC29/WG11 M31520, MPEG*, October 2013.

[6] E. Bondarev, S. Zinger, and P. H. N. de With, "Performance-efficient architecture for free-viewpoint 3DTV receiver," in *2010 Digest of Technical Papers International Conference on Consumer Electronics (ICCE)*, Jan 2010, pp. 65–66.

[7] L. Do, G. Bravo, S. Zinger, and P. H. N. de With, "Real-time free-viewpoint DIBR on GPUs for large base-line multi-view 3DTV videos," in *2011 Visual Communications and Image Processing (VCIP)*, Nov 2011, pp. 1–4.

[8] A. Akin, R. Capoccia, J. Narinx, J. Masur, A. Schmid, and Y. Leblebici, "Real-time free viewpoint synthesis using three-camera disparity estimation hardware," in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2015, pp. 2525–2528.

[9] http://vision.middlebury.edu/stereo/.

[10] K. Takahashi, "View interpolation sensitive to pixel positions," in *2013 IEEE International Conference on Image Processing*, Sept 2013, pp. 2207–2211.

[11] Y. Li, K. Huang, and L. Claesen, "Soc oriented real-time high-quality stereo vision system," in *2016 IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, Sept 2016, pp. 1–6.

[12] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, April 2004.

[13] G. Fhr, G. P. Fickel, L. P. Dal'Aqua, C. R. Jung, T. Malzbender, and R. Samadani, "An evaluation of stereo matching methods for view interpolation," in *2013 IEEE International Conference on Image Processing*, Sept 2013, pp. 403–407.