View Synthesis using Depth Map for 3D Video

Cheon Lee and Yo-Sung Ho
Gwangju Institute of Science and Technology (GIST)
1 Oryong-dong, Buk-gu, Gwangju, 500-712, Republic of Korea
E-mail: {leecheon, hoyo}@gist.ac.kr Tel: +82-62-970-2258

***Abstract*— Three-dimensional (3D) video involves stereoscopic or multi-view images to provide depth experience through 3D display systems. Binocular cues are perceived by rendering proper viewpoint images obtained at slightly different view angles. Since the number of viewpoints of the multi-view video is limited, 3D display devices should generate arbitrary viewpoint images using available adjacent view images. In this paper, after we explain a view synthesis method briefly, we propose a new compensation algorithm for view synthesis errors around object boundaries. We describe a 3D warping technique exploiting the depth map for viewpoint shifting and a hole filling method using multi-view images. Then, we propose an algorithm to remove boundary noises that are generated due to mismatches of object edges in the color and depth images. The proposed method reduces annoying boundary noises near object edges by replacing erroneous textures with alternative textures from the other reference image. Using the proposed method, we can generate perceptually improved images for 3D video systems.**

I. INTRODUCTION

Considerable improvements of information technologies (IT) enable us to enjoy diverse forms of multimedia services. Recently, 3D video becomes an emerging medium that can provide more realistic and natural experiences to users by auto-stereoscopic displays or free viewpoint TV (FTV). One of main challenges is rendering continuous viewpoint images using 3D video. Thus, the image interpolation method to generate arbitrary viewpoint images using the multi-view video is a key part of 3D display systems.

Since stereoscopic images were introdued by Charles Wheatstone [1], there have been a lot of interesting proposals for three-dimensional video (3DV). Since December 2001, the moving picture experts group (MPEG) has investigated the 3D audio-visual (3DAV) format. Focusing on the multi-view video, MPEG and joint video team (JVT) have recently developed an international standard on multi-view video coding (MVC) [2].

As a second phase of FTV efforts, MPEG has started a new work on 3D video coding since April 2007. It basically employs multi-view video and its corresponding depth data; thus, it provides functionalities for free viewpoint navigation and 3D display [3]. In order to perform exploration experiments, the MPEG 3DV group has collected 10 test sequences and examined validity of sequences by viewing on 3D display systems in April 2008 [4].

The input data of 3DV system consists of multi-view video sequences and their associated depth data. The multi-view video is captured by multiple camera arrays to provide wide angle of view. The depth data can be generated by both the computer-vision-based depth estimation method or with the time-of-flight (TOF) depth camera. The first approach compares pixel similarities between multi-view images according to disparity values. The second approach measures the reflection time of rays. In this work, we mainly focus on the computer-vision-based algorithm.

Since the 3DV system involves more than two images, the amount of data increases proportional to the number of cameras. Hence, the 3DV system needs an efficient video codec. After compressing the 3D video data, the decoder reconstructs the coded data and renders a 3D scene by selecting stereoscopic or multi-view images. If the rendering device needs arbitrary intermediate viewpoint images, we need to generate them using the view synthesis method in conjunction with the decoded depth data.

The 3D warping technique in image-based rendering (IBR) is an appropriate method for arbitrary viewpoint rendering. It generates a nearby viewpoint image by projecting a point from the reference view to the virtual view using the depth data [6]. If we estimate the depth data using a software-based method, the hole problem occurs due to occlusion and disocclusion. Handling the hole problem is one of the difficult tasks in IBR. We explain the view synthesis method and a solution to the hole problem. When we generate the intermediate view image using depth information, we obtain a synthesized image containing boundary noises generated by inaccurate depth values around object edges. We propose a boundary noise removal algorithm for the 3D video system.

The rest of this paper is organized as follows. In Section II, we briefly explain the concept of the 3D video system. In Section III, we describe the view synthesis method using depth data. In Section IV, we define the boundary noise and propose the removal method. In Section V, improvement of synthesized images using the proposed algorithm is represented. Finally, this paper is completed with our conclusions presented in Section VI.

II. 3D VIDEO SYSTEM

Figure 1 shows a general 3D video system [3] whose inputs are multi-view video sequences and associated depth data. After obtaining the 3D video data, we compress it using a new codec and transmit the bitstream to the channel. Then, the receiver reconstructs all coded data and renders proper view images on the 3D display monitor. If the number of decoded views is less than that of necessary views for the 3D displays, the rendering device needs to generate more views using the view interpolation method.

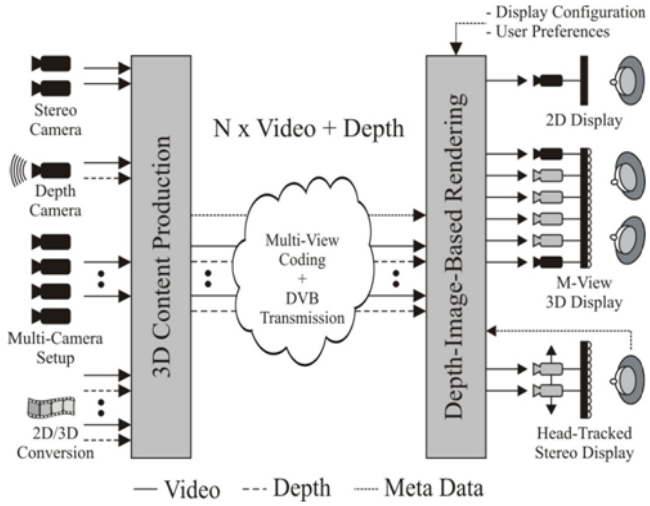


Fig. 1. 3D Video System

A. *Capturing Multi-view Video Data*

The 3D video is reproduced by the multi-view video captured by multiple cameras. In order to develop the experimental environments, the subgroup for 3D video coding in MPEG has directed requirements on capturing 3D test materials [7].

1) *Camera arrangement*: the multi-view camera should be 1-D parallel having uniform camera interval 5 ~ 6.5 cm. Other types of camera arrangement are possible, but we are focusing on only the 1-D parallel case.
2) *Rectification*: all videos should be rectified to provide parallel views. With rectified multi-view images, we can find the corresponding pixels at the same height of images.
3) *Synchronization*: accurate temporal synchronization of the multiple cameras is inevitably required.
4) *Color consistency*: although all cameras are of the same type of model, color consistency between views can be easily broken by inaccurate camera settings. Therefore, all videos should be corrected by a proper color correction method.
5) *Camera calibration*: accurate camera parameters by an accurate calibration method are inevitably required. Those parameters explain the geometrical information of the cameras.
6) *Content*: since the goal of 3D video is the standardization of a new codec, contents should be proper for developing good coding algorithms. Sufficient texture variation, moving objects, and complex depth structure should be considered in the scene.

Experts of the subgroup of 3D video coding consider only the 1-D parallel multi-view camera array. Although all 1-D parallel cameras are pointing the same direction, the captured images are not rectified, as shown in Fig. 2(a). If we use those images as input for stereoscopic displays, we hardly feel depth impression. Therefore, we should rectify captured images, as shown in Fig. 2(b) [8][9].

(a) Unrectified case

(b) Rectified case

Fig. 2. Overlapped images of three viewpoint images

Another important requirement for the 3D test data is color consistency between neighboring views. Even if the use of same camera, it does not guarantee color consistency with different views. Color mismatches between adjacent views deteriorate the performance of several methods, such as depth estimation, view synthesis, and video compression. Hence, we need to correct color inconsistency of the 3D test data.

B. *Generation of Depth Map*

The 3D video system utilizes the depth data for generation of arbitrary view images. Conceptually, the depth value describes distance from the camera to objects in the scene. There are mainly two approaches to obtain the depth data. The hardware-based depth estimation method, such as TOF depth camera, uses a sensor measuring the reflection time of rays. It provides quite accurate depth data, but it has some constraints: the scene should be captured indoor and hair should be avoided since it scatters rays. On the other hand, the software-based method has no constraint except for image rectification, while accuracy of the estimated depth map is quite unstable around object boundaries due to occlusion and disocclusion. Handling occlusion and disocclusion is the well-known issue in depth estimation.

Currently, the 3D video subgroup in MPEG is testing the depth estimation software [10]. It uses two reference images, left and right views, to generate intermediate depth images. Basically, it employs a block matching method to calculat the cost function, and applies graph cut algorithm for disparity refinement [11]. Moreover, we can cooperate with other methods, such as ‘sub-pel precision’ and ‘temporal enhancement’ [12][13]. The ‘sub-pel precision’ method upsamples reference images as much as 2 or 4 times in the horizontal direction to estimate fine and accurate depth values. The ‘temporal enhancement’ method reduces flickering artifacts in the depth video by determining the static area.

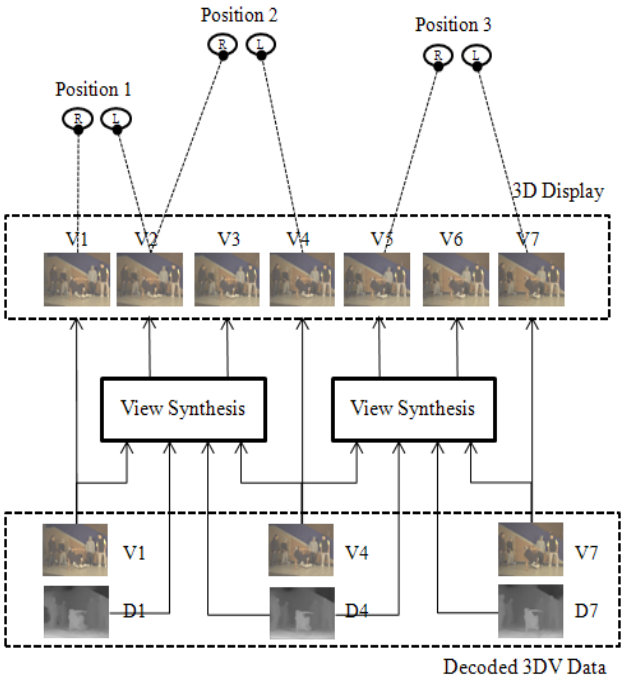


Fig. 3. 3D video rendering with view synthesis

C. 3D Video Coding

The main objective of the MPEG activity of 3D video coding is the standardization of a new codec dealing with the 3D video data. The new standard of 3D video will enable stereo devices to cope with various display types and sizes, and different viewing preferences [14].

In order to achieve the vision on 3D video, they are considering multi-view video sequences and associated depth data as inputs because those data can provide a wide angle of the scene. However, since the amount of the input data increases proportional to the number of cameras, an efficient coder for multi-view video and its depth data is necessary. Currently, the format of 3D video is not yet determined, but the depth data should be encoded simultaneously with the multi-view video. In addition, an important requirement for the new standard is compatibility with existing standards, such as H.264/AVC and MVC.

D. 3D Video Rendering

After reconstructing the 3D video data at the receiver, the 3D display device should render proper images on the monitor. For example, if the type of display is a stereoscopic device, the display device selects two viewpoint images among the reconstructed multi-view images. Moreover, if the device is the N-view auto-stereoscopic display, it should select N-view images. If the number of the reconstructed views is less than N, the decoder should generate more views using a view interpolation method. Similarly, Fig. 3 explains the functionality of free-view navigation. If the 3D display device has only three reconstructed views and associated depth data, it should generate more intermediate images for natural 3D display.

III. View Synthesis using Depth Map

In this section, we explain the procedure of the view interpolation method. Among various techniques of IBR, we use the 3D warping technique because it utilizes depth information for viewpoint shifting. The depth value describes the distance between the camera and objects in the scene. Using this geometrical information, we can map corresponding pixels between different viewpoints. When we change the viewpoint, some background regions are disappeared or appeared because of foreground objects. This induces the hole problem. After we explain the 3D warping technique, we explain the hole problem and propose its solution in detail. The whole procedure of the view synthesis method is presented in Figure 4.

A. Pixel Correspondence using Depth Map

Camera parameters describe the relationship between camera coordinates and world coordinates. They consist of one intrinsic parameter **A** and two extrinsic parameters: the rotation matrix **R** and the translation vector **t**. If a point **X** in the real world is projected to the pixel **x** of an image, we can formulate it using the projective matrix **P**=**A**[**R**|**t**].

Assuming that all multi-view cameras are calibrated, we can define the pixel correspondences for all images captured by multiple cameras. When a point $\tilde{\mathbf{M}}$ in the world coordinates is projected to the camera coordinates, a pixel $\tilde{\mathbf{m}}$ in the image can be found by

$$\tilde{\mathbf{m}} = \mathbf{P}\tilde{\mathbf{M}} = \mathbf{A}[\mathbf{R} \mid \mathbf{t}]\tilde{\mathbf{M}} \tag{1}$$

where a single point $\tilde{\mathbf{M}} = [X \quad Y \quad Z \quad 1]^T$ in world coordinates and a projected point $\tilde{\mathbf{m}} = [x \quad y \quad 1]^T$ represents form of the homogeneous pixel position.

When we find corresponding pixels between the reference and target viewpoint images, we can put a pixel $\mathbf{m_r}$ in the reference image back to the world coordinates by

$$\mathbf{M_r} = \mathbf{R_r}^{-1} \cdot \mathbf{A_r}^{-1} \cdot \mathbf{m_r} \cdot \mathbf{d}(\mathbf{m_r}) - \mathbf{R_r}^{-1} \cdot \mathbf{t_r} \tag{2}$$

where $\mathbf{A_r}$, $\mathbf{R_r}$, and $\mathbf{t_r}$ represent camera parameters of the reference view, and $\mathbf{d}(\mathbf{m_r})$ is a depth value for the position of $\mathbf{m_r}$ in the reference image. After this backward projection, we project $\mathbf{M_r}$ into coordinates of the virtual camera using Eq. (3). As a result, we can find the relationship between two positions $\mathbf{m_r}$ and $\mathbf{m_t}$ as follows.

$$\tilde{\mathbf{m}}_\mathbf{t} = \mathbf{A_t}[\mathbf{R_t} \mid \mathbf{t_t}]\tilde{\mathbf{M}}_\mathbf{r} \tag{3}$$

A popular format for the depth information is the 8-bit single channel image. Hence, the range of representing the depth of field is from 0 to 255. It means that the number of depth planes is 256 at most. Using this limitation, we can find 256 homography matrices by defining pixel correspondences between adjacent views. A homography matrix is formed by a 3x3 matrix having eight degree of freedom (DOF); hence we

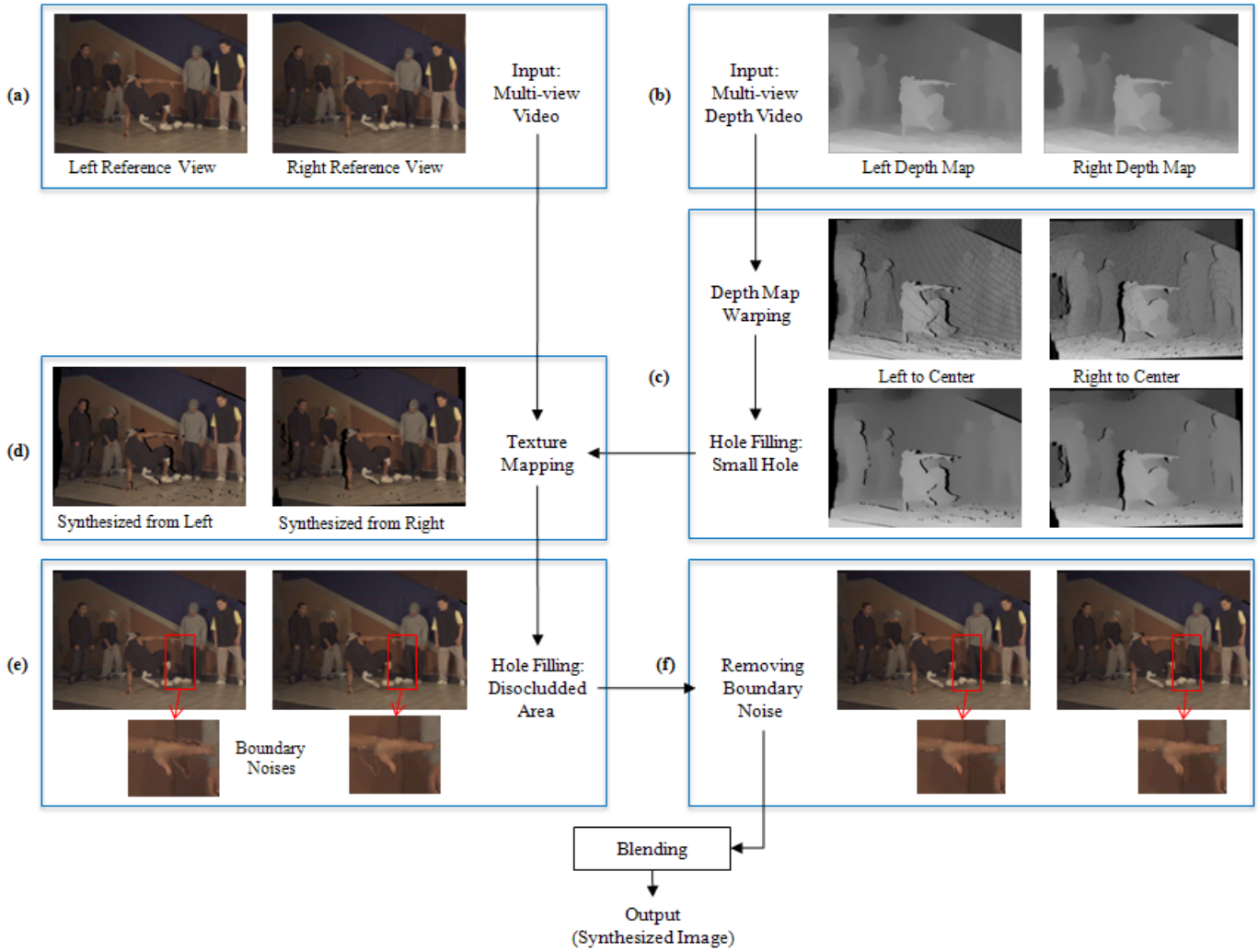


Figure 4. Whole procedure of view synthesis: (a) original reference images, (b) depth maps corresponding two reference images, (c) generated depth maps using 3D warping and small hole filling, (d) synthesized images on the virtual viewpoint, (e) hole filled images by referring to the other synthesized image, (f) noise removed images using the proposed method.

can determine it with four corresponding pixels for one depth level. Equation (4) explains relationship between two corresponding pixels with a homography matrix. $\mathbf{H_d}$ represents the homography matrix for a depth value **d.**

$$\tilde{\mathbf{m}}_t = \mathbf{H}_d \tilde{\mathbf{m}}_r \tag{4}$$

B. Depth Map Warping

Since the 3D warping technique uses corresponding pixels between neighboring views, we can synthesize a virtual image by mapping all pixels between the target view and reference view images. Practically, we use a different trick to map the pixel correspondences between views using the depth map warping. Once we obtain the depth map of the target view using the 3D warping method by exploiting the reference depth map itself, we set pixel correspondence between the target and reference views in conjunction with the warped depth map. During this process, small holes due to viewpoint shifting are filled using a median filter. This trick reduces effects of small holes generated by the rounding operation during viewpoint shifting. Figure 5(c) shows both the warped depth map having small holes and the hole-filled depth map containing only wide holes painted in black.

C. Texture Mapping

The next step is texture mapping for the target viewpoint image. Using 3D warping, we obtained depth maps warped from two reference views. Hence, we can find the mapped pixels between two reference images and one target image. With this method, we can map colors of the target view image. If the calculated position is not a full-pel position in the reference view, we interpolate the pixel intensity using neighboring pixel values. Resultant images are represented in Fig. 5(d), where there are no small holes. Black areas are the newly exposed regions which have no texture in the reference image. We fill in those regions by the next process.

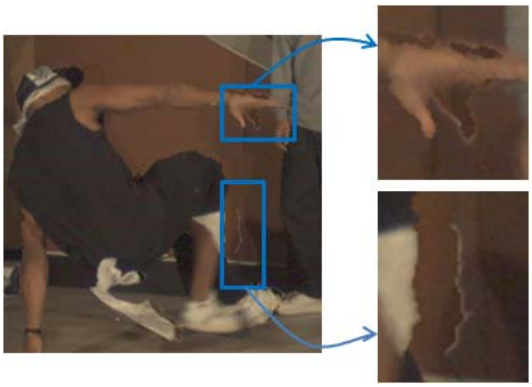


Fig. 5. Boundary Noise around Depth Discontinuity

D. *Hole Filling*

The hole area is the newly exposed region generated by viewpoint shifting. Specifically, when the viewpoint is shifted from left to right in the horizontal direction, occluded regions in the left image appear in the right image around the right side of the object. If we have only one reference image and its depth map, we cannot fill holes with textures, except for inpainting with neighboring textures. However, since we use multi-view images and their depth maps, we can easily find corresponding textures from other reference images for the hole region. Using this method, we can fill in the hole area, as shown in Fig. 4(e).

IV. BOUNDARY NOISE REMOVAL

We obtained the hole-filled images referring, to both the left and right views. If the depth map is accurate, we do not need any further process. However, since we use the depth map estimated by a software-based method, we should eliminate boundary noises generated by depth mismatches around object boundaries. In this section, we define the boundary noise and propose a boundary noise removal method.

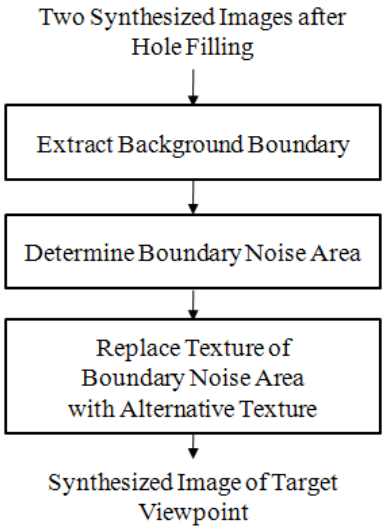


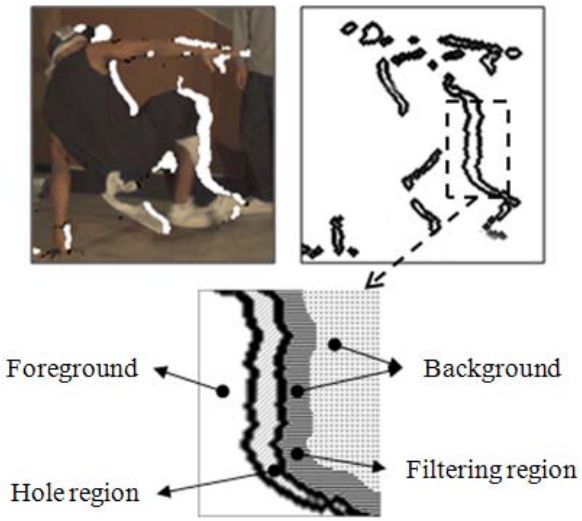
Fig. 6. Procedure of Boundary Noise Removal

A. *Boundary Noises around Depth Discontinuity*

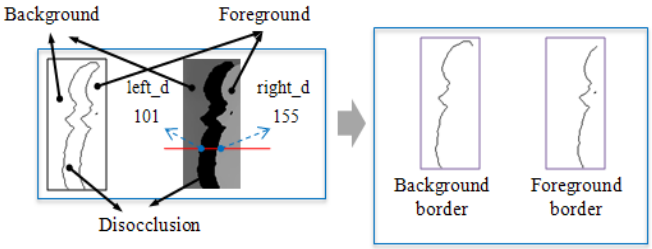
The performance of the view synthesis method using 3D warping is highly dependent on accuracy of depth information. Hence, false depth values around depth discontinuities create boundary noises. Figure 5 shows an example image of boundary noises which are distributed in the background region near objects. It is because of inaccurate depth values around object boundaries. If a depth value of the foreground object is similar to that of the background, the texture of the foreground object is mapped to the background region in the target view image. When a user sees these noises through the 3D display, he or she may feel uncomfortable. Hence, we need to eliminate such noises in the synthesized image for more natural 3D image display.

B. *Proposed Boundary Noise Removal Method*

Figure 6 explains the whole procedure of the proposed method. Since boundary noises appear in the background area near object boundaries in the target view, we can solve the problem by processing background textures. In order to detect the background, we check significant depth changes and determine the background border adjacent to foreground objects. Along the object border, we set the target region. Then, we replace noise textures in the target image by texture of the other reference image.



(a) Detecting boundary of hole segment



(b) Extracting background boundary

Fig. 7. Extracting Background Border

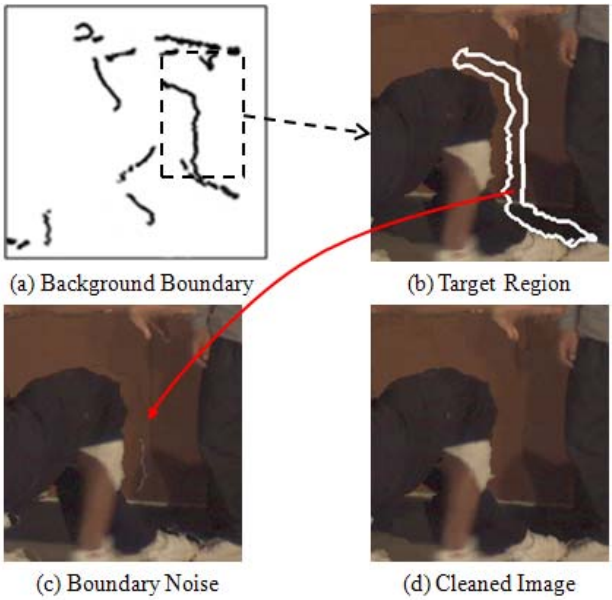


Fig. 8. Boundary Noise Removal

C. *Detection of Background Boundary*

Although the boundary noise appears around object boundaries, we do not consider gradually changing depth values because those depth differences do not generate boundary noises. Hence, we detect significant depth discontinuities using a predetermined depth threshold value. Practically, we determine the contour of hole regions, as illustrated in the left image in Fig. 7(a). The top right image of Fig. 7(a) is the determined hole boundary.

Using the contour, we divide the synthesized image into three regions: the foreground region, the hole region, and the background region. Figure 7(b) explains steps of extracting the background contour. After defining the boundary contour, we check the depth difference of two valid depth values and decide which depth is located at the background. In Fig. 7(b), if the left depth value is 101 and the right depth value is 155, the left one is located at the background: hence, we decide that the left pixel is in the background contour. In this manner, we can obtain the background boundary, as shown in the right image of Fig 7(b).

D. *Boundary Noise Removal*

After determining the background contour along the hole regions, we determine to filter target regions, as painted in gray in Fig. 7(a). Toward the opposite direction of the hole region, we determine the filtering region along the background border using a predetermined width. The filtering region should not overlap the foreground region.

Using filtering regions, we find corresponding textures in the other reference image. Since we are using the multi-view video, the alternative texture information exists in the other reference view. For filtering regions, we copy the alternative information from the other reference image. Replacing textures of the target image with alternative textures, we obtain the final synthesized image, as shown in Fig. 8(d).

V. Experimental Results and Discussions

We used two metrics to evaluate the performance of the proposed method: PSNR value and viewing subjective perception. The main goal of our research is to improve visual quality of the synthesized image, thus we compared quality of two resultant images: an image without boundary noise, and the image generated by the conventional method. As an objective measure, we calculated PSNR values for all synthesized images. In order to calculate the PSNR value, we set the existing intermediate views to virtual viewpoints. For example, we generated an image at View 1 and View 2 using View 0 and View 3, and then we compared PSNR values between the original and generated images. We chose four sequences. Two sequences, 'Breakdancers' and 'Ballet', are provided by Microsoft Research. The other two sequences, 'Pantomime' and 'Lovebird1', are selected from the 3D video test sequences provide by MPEG. In the following subsections, we show results of synthesized images and their objective quantities.

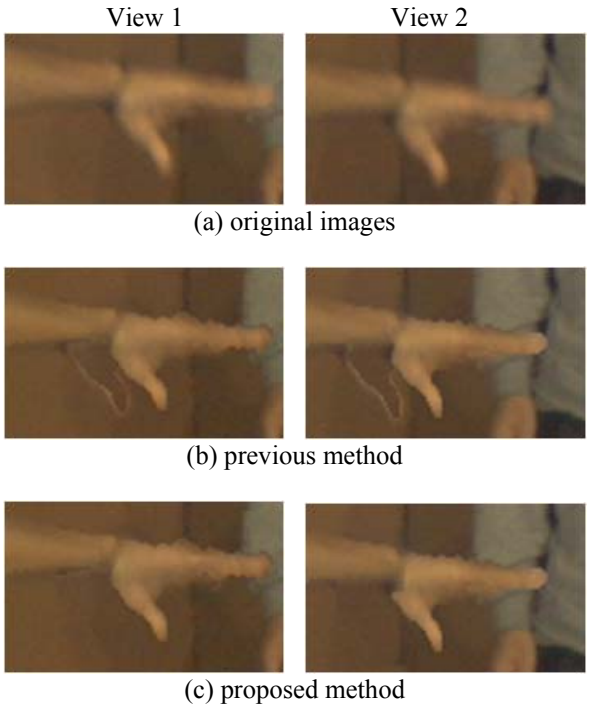
Table 1. Comparison of PSNR for Multi-view Sequences

Test Data	Previous method (A)		Proposed method (B)		Δ PSNR (C) = (B) – (A)	
	View1	View2	View1	View2	View1	View2
Breakdancers	30.361	31.692	30.428	32.076	0.067	0.384
Ballet	25.060	25.521	25.101	25.574	0.042	0.053

(Unit: dB)

A. *Experimental Results on Multi-view Video Sequences*

Microsoft Research has provided multi-view video sequences and their depth maps. The depth map is generated

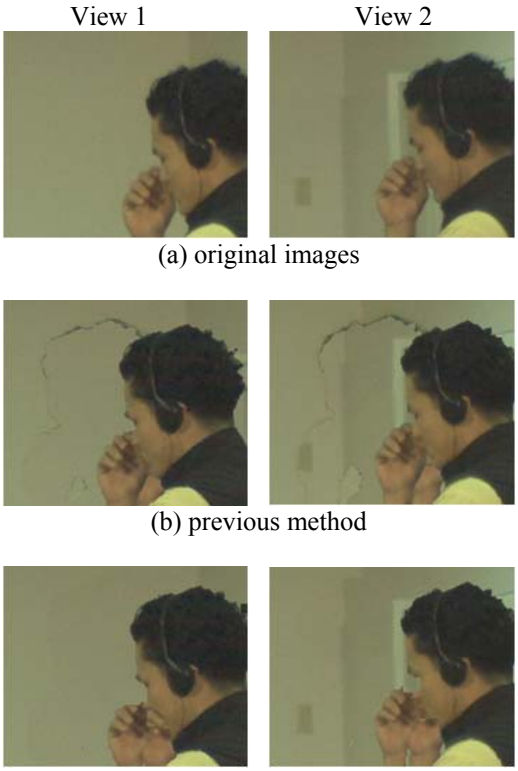


(a) original images

(b) previous method

(c) proposed method

Fig. 9. Synthesized Images of 'Breakdancers'



(a) original images

(b) previous method

(c) proposed method

Fig. 10. Synthesized Images of 'Ballet'

by the segment-based depth estimation method [15]. The file size is 1024x768, and the multiple camera rig is arc with eight cameras. All views are not rectified but calibrated. Among eight views, we selected two reference views to generate intermediate view images. The number of total synthesized frames is 100.

Table 1 shows average PSNR values for each virtual view. As you can notice, the PSNR values are similar except for View 2 of the 'Breakdancers' sequence, which implies that the proposed algorithm does not degrade the quality of the resultant image. Figure 10 shows comparison of visual quality of the 'Breakdancers' sequence. You can easily observe boundary noises around the thumb in Fig. 9(b). These noises are generated by mismatches of depth boundaries in the depth map comparing to that of the color image. After processing the proposed method during the view synthesis method, boundary noises have been removed clearly, as shown in Fig. 9(c).

Similarly with the 'Breakdancers' sequence, the 'Ballet' sequence has the same problem in the depth image. Figure 10 shows synthesized images. When we use the previous method, some boundary noises of the man's hair appeared on the wall. However, the proposed method removed those noises from the synthesized images, as shown in Fig. 10(c).

B. Experimental Results on 3D Video Test Sequences

Since there is no available depth data, we need to generate the depth video using the provided depth estimation reference software (DERS) [16]. Currently, MPEG has developed both the depth estimation software and the view synthesis software as auxiliary softwares for the 3D video system. Several algorithms, such as the 'sub-pel precision' and the 'temporal enhancement', have been integrated in the software. We used the 'half-pel precision' and 'temporal enhancement' method when we generate depth maps.

The 'Pantomime' sequence is provided by Nagoya University [17]. It is captured by 80 cameras with a 1-D parallel camera rig. All views are rectified and color corrected. The file size is 1280x960, and the number of frames is 500. All cameras are rectified and color corrected, and all parameters are provided. Another test sequence is the 'Lovebird1', which is provided by ETRI and MPEG Korea [18]. It consists of rectified 12 views with camera parameters. The file size is 1024x768.

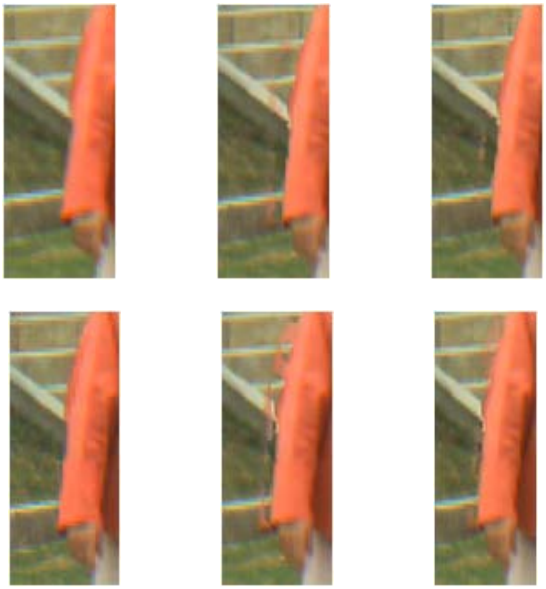
Table 2. Comparison of PSNR for 3D Video Sequence

Test Data	Previous method (A)		Proposed method (B)		Δ PSNR (C) = (B) – (A)	
Pantomime	View39	View40	View39	View40	View39	View40
	35.112	34.568	35.075	35.488	-0.036	0.920
Lovebird1	View6	View7	View6	View7	View6	View7
	31.362	31.646	31.356	31.302	-0.006	-0.344

(Unit: dB)

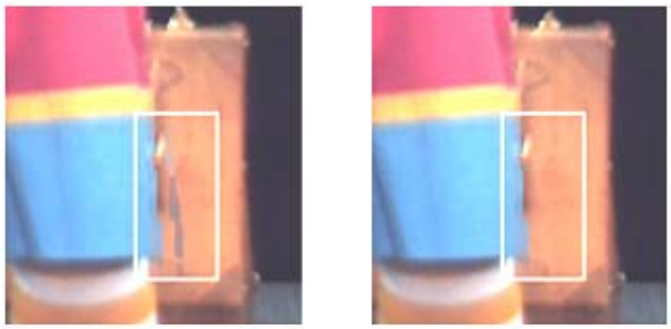
Table 2 shows comparison of PSNR values of resultant images. Both sequences showed relatively high view synthesis results. In case of View 40 of the 'Pantomime' sequence, the proposed method improved quality of the image by 0.92 dB. Other delta PSNR values are very small while subjective quality has been improved slightly.

Figure 11 shows comparison of synthesized images of the 'Lovebird1' sequence. This sequence has much boundary noises around the man's right arm, as shown in Fig. 11(b).



(a) (b) (c)

Fig. 11. Synthesized Images of 'Lovebird1': (a) original images (b) previous method (c) proposed method



(a) previous method (b) proposed method

Fig. 12. Synthesized Images of 'Pantomime'

After applying the proposed method to the view synthesis algorithm, we obtained enhanced results, as shown in Fig. 11(c). In Fig. 12, we compare synthesized images of the 'Pantomime' sequence. As you can see, the boundary noises are generated around the trouser. After applying the proposed algorithm, we obtained the noise removed image, as shown in Fig. 13.

Generally, problems in view synthesis are caused by depth estimation errors around depth discontinuity. In other words, if depth estimation performs well on the area, the boundary noises will not appear in the synthesized image. However, due to the occlusion region, this problem is inevitable. Therefore, the proposed algorithm can be a good error compensation tool for the 3D video system preventing boundary noises.

VI. Conclusion

Two key techniques for view synthesis are 3D warping and hole filling methods. In this paper, we have described the procedure of the view synthesis method which generates intermediate view images using depth information. We also proposed the boundary noise removal method eliminating boundary noises from the synthesized image around object boundaries. Since the number of transmitted views is limited due to channel capacity, the 3D video system employs the view synthesis method to generate intermediate images for 3D rendering. In addition, we proposed the boundary noise removal method. After we detect the target region using the provided depth map, we determine the target region where we replace textures of the synthesized image with alternative textures referring to the other reference image. For experiments, we used four multi-view sequences and compared visual perception of resultant images. In order to check objective quality, we compared PSNR values on resultant images. Image quality is perceptually improved, while boundary noises are removed. In some cases, the PSNR value was increased by 0.92 dB when we applied the proposed filtering method.

Acknowledgment

This research was supported by the MKE, Korea, under the ITRC support program supervised by the IITA (IITA-2009-C1090-0902-0017)

References

[1] C. Wheatstone, "Phenomena of Binocular Vision," Philosophical Transactions of the Royal Society of London, 371-394, 1838.

[2] JVT of ISO/IEC MPEG & ITU-T VCEG, JVT-AD207, "WD 4 Reference software for MVC," Feb. 2009.

[3] ISO/IEC JTC1/SC29/WG11, "Introduction to 3D Video," N9784, May 2008.

[4] ISO/IEC JTC1/SC29/WG11, "Description of Exploration Experiments in 3D Video Coding," N9783, May 2008.

[5] ISO/IEC JTC1/SC29/WG11, "Results of 3D Video Expert Viewing," N9992, July 2008.

[6] L. McMillan, "An Image-based Approach to Three-dimensional Computer Graphics," Technical report, Ph.D. Dissertation, UNC Computer Science TR97-013, 1999.

[7] ISO/IEC JTC1/SC29/WG11, "Call for Contributions on 3D Video Test Material," N9595, Jan. 2008.

[8] R. Hartley, "Theory and Practice of Projective Rectification," International Journal of Computer Vision, vol. 35, no. 2, 1999.

[9] Y.S. Kang, and Y.S. Ho, "Geometrical Compensation Algorithm of Multi-view Images for Arc Multi-Camera Array," LNCS, 5353, pp. 543-552, 2008.

[10] ISO/IEC JTC1/SC29/WG11, "Reference Softwares for Depth Estimation and View Synthesis," m15377, April 2008.

[11] Y. Boykov and V. Kolmogorov, "An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision," Proc. International Workshop Energy Minimization Methods in Computer Vision and Pattern Recognition, pp. 359-374, Sept. 2001.

[12] ISO/IEC JTC1/SC29/WG11, "Experimental Results on Depth Estimation and View Synthesis with subpixel-precision," m15584, July 2008.

[13] ISO/IEC JTC1/SC29/WG11, "Experimental Results on Improved Temporal Consistency Enhancement," m16063, Jan. 2009.

[14] ISO/IEC JTC1/SC29/WG11, "Vision on 3D video," N10357, Jan. 2009.

[15] L. Zitnick, S.B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality Video View Interpolation using a Layered Representation," SIGGRAPH 2004, Aug. 2004.

[16] ISO/IEC JTC1/SC29/WG11, "Depth Estimation Reference Software (DERS) 4.0," M16605, July 2009.

[17] ISO/IEC JTC1/SC29/WG11, "1-D Parallel Test Sequences for MPEG-FTV," M15378, April 2008.

[18] ISO/IEC JTC1/SC29/WG11, "Contribution for 3D Video Test Material of Outdoor Scene," M15371, April 2008.