Dual Edge-Confined Inpainting of 3D Depth Map Using Color Image's Edges and Depth Image's Edges

Ming-Fu Hung[†], Shaou-Gang Miaou[*†], and Chih-Yuan Chiang[†]
[†]Master Program in Communication Engineering, Chung Yuan Christian University, Taoyuan, Taiwan, R.O.C.
[*]Department of Electronic Engineering, Chung Yuan Christian University, Taoyuan, Taiwan, R.O.C.
E-mail: miaou@cycu.edu.tw Tel: +886-3-2654609

***Abstract*—Currently, most 3D formats adopt the approach of 2D plus depth map. Thus, many papers discuss depth maps, including how to inpaint them. When inpainted, the main focus is on edge areas because poor edges in depth maps will result in two problems: (1) There will be holes in 3D images after image synthesis; (2) When synthesized for other views, an edge mismatch may create some visual defect at the occlusion part of the images. In this paper, we propose a method to enhance the consistency between a color image and its depth map. With 2D color image's edges and depth map's edges, the proposed method finds the areas where the edges do not match, and uses a dual edge-confined inpainting technique to inpaint the edge-mismatched areas. The inpainting technique exploits the relationship between edges and objects in color image and depth map, and conducts inpainting directionally, where a depth map is inpainted at the inconsistent part between a depth map and its corresponding color image to improve the quality of the depth map. The experimental results show that the proposed method enhances the edge consistency between the depth map and the corresponding color image. In addition, following traditional de-blocking process in image coding, the proposed method increases the PSNR value of depth map for QP less than 30, and achieves a comparable performance to that obtained by using the trilateral filter plus SD algorithm which also uses color image and depth map information.**

I. INTRODUCTION

At the entrance of International Consumer Electronics Show of 2012, the 3D television wall occupied the whole entrance. Every monitor in the exhibition displayed 3D video. In Smart TV Chinese Forum held by NPD Display Search in May 2012, the evolution of 3D television and the development trend of auto-stereoscopy were explored as topics to obtain more information about technical analyses and marketing trends for 3D. Nevertheless, for the current development in 3D industry, the Vice President of NPD Display Search pointed out that, along with the promotion by vendors for 3D TV, consumers are getting more interested in 3D video. However, the contents and relevant services are still insufficient at this moment [1].

In fact, as early as 2008, BS11, a cable TV channel in Japan, had broadcasted 3D programs. Australia started the trial of 3DTV in 2010 for 2 months. Brazil is the first country in the world to freely broadcast wireless 3DTV programs. Sky Satellite TV also started broadcasting 3DTV channels in Britain in April, 2011. People in Britain can utilize the current Sky with HD TV box and 3D ready television set to watch FIFA World Cup displayed in 3D. Singapore also launched 3DTV tests from June 15th, 2010. The operators of wireless TV, cable TV, and IPTV cooperated to start the tests of 3DTV on multiplatform [2]. Obviously, all these countries take 3D broadcasting systems as the development goal in the future.

In terms of transmission, it is an unbearable loading to transmit 3D video with an existing transmission standard. Therefore, substitutive solutions are sought constantly to improve the transmission capacity for video contents by providing higher transmission bandwidth and better video compression methods. The current transmission standards that support high definition video are Wireless HD and WiGig based on 60-GHz millimeter wave, as well as the IEEE 802.11ad standard that is under planning [3-5]. Many standards currently being developed are for transmitting images in higher definition and 3D as well.

Currently, the 3D image synthesis technology is divided into two categories. One is the Image Based Rendering (IBR) [6]. This technology conducts rendering with depth analyses on color images. However, this technology has two major problems: (1) Switching view angles may generate 'holes' in 3D images; (2) The relative movement between scenes and objects may increase the computational cost significantly and slow down the processing speed. The second category of technology called the Depth Image Based Rendering (DIBR) is proposed to deal with the problems [7]. This technology is a synthesis method derived from stereoscopic systems. In this technology, every color image is associated with a depth image (2D + Depth), where the depth image represents the depth of scene. Color images and depth images are then combined with DIBR technology to form 3D images. This technology can display 3D images more perfectly than IBR. However, the performance of DIBR relies heavily on the completeness of depth images obtained. Black holes may be generated during the 3D image synthesis process if the quality of depth image is very poor. In addition, when we perform 3D image rendering for other view angles, the unmatched edges between color and depth images produce holes and defects in color images after image inpainting.

Several simple interpolation techniques have been proposed to deal with the inpainting problem mentioned above [8-10]. Unfortunately, this convenient approach often produces an undesirable blurring and zigzag effect on edges. Some special filters were proposed to reduce the effect [11-13]. The authors of [14] offer three useful perspectives on the inpainting

problem: (1) (1) A depth map is often formed and expressed in grayscale, and the boundary edges of some objects are quite visible. (2) For a visible edge in a depth map, its corresponding edge is usually available in the color image associated with the depth map. (3) A depth map is usually not displayed directly for viewing, but is used in conjunction with the color image.

This study is aimed at the inpainting of the edges in depth images. The idea is to find the unmatched locations between a color image and its corresponding depth image and inpaint the unmatched locations, so as to increase the edge consistency between the color image and the depth image. The objective of this study is to enhance the 3D image quality with a proposed inpainting technique.

II. SYSTEM STRUCTURE

A system flowchart of the proposed inpainting method is shown in Fig. 1. The idea of the method is to find and inpaint the unmatched areas between a depth image and its corresponding color image. Firstly, edge detection will be conducted on the color image and the depth image. Then, locate the areas where edges in the two images are unmatched. Next, determine if the areas require inpainting, determine the direction of inpainting, and finally perform inpainting. More details on each block of Fig. 1 will be given later.

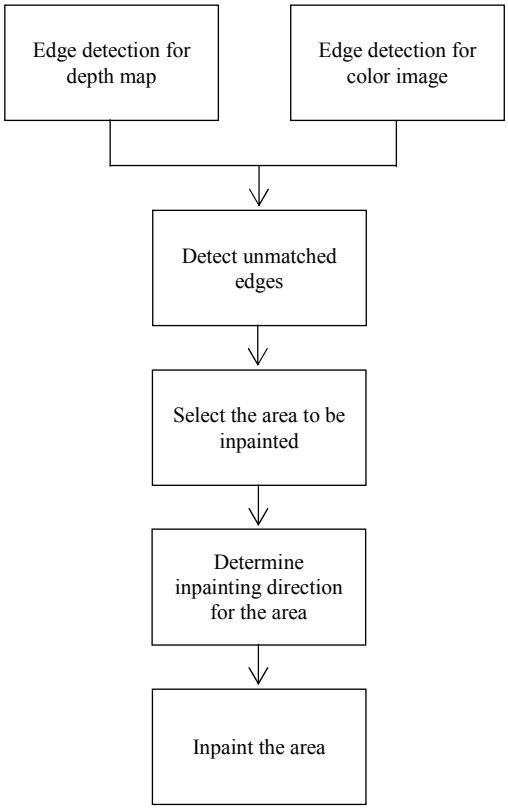


Fig. 1 A flowchart of the proposed inpainting method.

A. *Edge Detection*

This study utilizes edge information from both depth image and color image to find unmatched edge locations. Therefore, edge detection is required to find the edge information of interest. The major difficulties in performing successful edge detection come from the following issues:

1. Quantization errors and sensing noises cause obvious variation in brightness. These false edges may be detected, while some true edges may be misdetected because there is no significant variation in brightness.
2. An edge should present strong enough variation in brightness in order to be detected. This is the high frequency characteristics. Therefore, any smoothing operation that tries to reduce image noises will case signal blurring of edge area.

Fortunately, the edges in a typical depth image are quite obvious because the depth image presents depth information relatively coarsely in gray scale. It only reflects a rough distance between the object in a scene and the camera used to take the scene. Furthermore, it does not contain too much textures information as compared to a color image. However, we not only consider the edge of depth image but also the edge of color image. Therefore, how to choose a better way to do edge detection will also affect the performance of our approach.

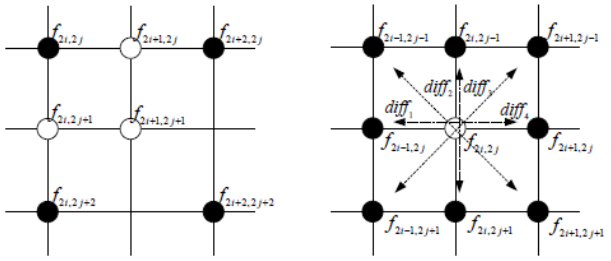
The Canny edge detector is selected here [15] for three advantages summarized as follows:

1. Good detection capability: able to find all edge lines with enough variation in gray scale and there are no false boundary points. The detection accuracy is high when applied to a depth image, and the false edges are less when applied to a color image.
2. Good edge positioning: able to make the detected boundary positions close to the positions of real edges. The position error is usually small enough for many applications, including ours. This advantage is important for us because we need to find the unmatched edges between color image and depth image and poor edge location accuracy may reduce the performance of our approach.
3. Good response values: when two corresponding edge images are checked, the edge lines on an inpainted area must be thin enough such that the inpainted area will not be occluded by the edges detected.

An image interpolation method is usually used to reduce the distortion effect induced by image zooming. In the context of 3D imaging, the interpolation method is usually utilized to fill holes. A fast edge interpolation algorithm is introduced next.

B. *Fast Edge Interpolation, FEI*

This interpolation method divides an image into smooth pixel areas and edge pixel areas. Then linear interpolation is applied to smooth pixels and non-linear interpolation is applied to edge pixels. Fig. 2 shows the relevant pixel arrangement for the FEI method. The FEI method will be used later for inpainting depth images.



(a) (b)

Fig. 2 A schematic showing FEI method. (a) Interpolation for smooth pixels; (b) Interpolation for edge pixels.

The procedure of the FEI method is given as follows:

Step0. Le $f_{i,j}$ be a pixel value at coordinate (i, j) and T be a threshold. $p, q \in \{(0, 1), (1, 0)\}$.

Step1. Referring to Fig. 2, calculate edge strength.

$$\Delta f_1 = \left| f_{2i,2j} - f_{2i+2p,2j+2q} \right|$$

$$\Delta f_2 = \left| f_{2i+2,2j} - f_{2i,2j+2} \right|$$

$$\Delta f_3 = \left| f_{2i,2j} - f_{2i+2,2j+2} \right|$$

Step2. Interpolation for smooth pixels (Fig. 2(a)).

$$\textit{if } \Delta f_1 < T \textit{ then } f_{2i+2p,2j+2q} = (f_{2i,2j} - f_{2i+2p,2j+2q})/2$$

$$\textit{else } f_{2i+2p,2j+2q} \text{ is an edge point}$$

$$\textit{if } \Delta f_2 < T \textit{ and } \Delta f_3 < T \textit{ then } \Delta f_{\min} = \min\{\Delta f_2, \Delta f_3\}$$

$$\textit{if } \Delta f_{\min} = \Delta f_2$$

$$f_{2i+1,2j+1} = (f_{2i+2,2j} + f_{2i,2j+2})/2$$

$$\textit{else } f_{2i+1,2j+1} = (f_{2i,2j} + f_{2i+2,2j+2})/2$$

$$\textit{else if } \Delta f_2 < T \textit{ then } f_{2i+1,2j+1} = (f_{2i+2,2j} + f_{2i,2j+2})/2$$

$$\textit{else if } \Delta f_3 < T \textit{ then } f_{2i+1,2j+1} = (f_{2i,2j} + f_{2i+2,2j+2})/2$$

$$\textit{else } f_{2i+1,2j+1} \text{ is an edge point.}$$

Step3. Interpolation for edge pixels (Fig. 2(b)).

This part of the method is similar to that for smooth pixels. The differentiation amounts in horizontal, vertical, and diagonal directions are calculated within a 3×3 area. Take the average of two points that have the smallest differentiation amount to be the interpolated value, as shown in Figure 2(b). The relevant calculations are given as follows:

$$diff_1 = \left| f_{2i-1,2j} - f_{2i+1,2j} \right|$$

$$diff_2 = \left| f_{2i-1,2j-1} - f_{2i+1,2j+1} \right|$$

$$diff_3 = \left| f_{2i,2j-1} - f_{2i,2j+1} \right|$$

$$diff_4 = \left| f_{2i+1,2j-1} - f_{2i-1,2j+1} \right|$$

$$diff_{\min} = \min\{diff_k\}, \text{ for } 1 \le k \le 4$$

III. AREA SELECTION AND DUAL EDGE INPAINTING

In this section, we discuss the inpainting of edges for depth image. The inpainting process includes three major steps: (1) Edge removal and combination, (2) selecting inpainting areas, and (3) dual edge inpainting. An unmatched edge is inpainted with these three steps.

A. Edge Removal and Combination

We find the areas that require inpainting with the combination and removal of edges. Following the use of Canny edge detection, we make a binarized graph by marking a '1' for an edge pixel and a '0' for a non-edge pixel. Let the resulting binarized graph for a color image and the corresponding depth image be I_C and I_D, respectively. We then perform the Exclusive-OR operation on I_C and I_D:

$$I_C \oplus I_D = I_{XOR} \tag{1}$$

A '1' at location (i, j) in I_{XOR} indicates a disagreement of edge information between the color image and the depth image at that location. Given I_C and I_D, as well as I_{XOR}, with the logic AND operation (denoted by '*'), we obtain I'_C and I'_D:

$$I_{XOR} * I_C = I'_C \tag{2}$$

$$I_{XOR} * I_D = I'_D \tag{3}$$

where a '1' at location (i, j) in I'_C means the pixel at location (i, j) is an edge pixel in the color image while it is a non-edge pixel in the depth image at that location. Similarly, a '1' at location (i, j) in I'_D means the pixel at location (i, j) is an edge pixel in the depth image while it is a non-edge pixel in the color image at that location. Fig. 3 shows an example of I_C, I_D, I'_C, and I'_D.

Fig. 3(a) and 3(c) show the edge information of original color image and the original depth image, respectively, while Fig. 3(b) and 3(d) give the edge information of color image and depth image, respectively, but with their common edges removed. We can find that there are not many unnecessary edges remaining in the depth image. As for the color image, there are other texture objects that must be removed. Therefore, we compare the edge information in the 3 edge images, I_{XOR}, I'_C, and I'_D as follows. Each edge image contains lines that consist of edge pixels. Let l_{XOR} be a line in I_{XOR} and this line may consist of a line l_C in I'_C or a line l_D in I'_D or l_C and l_D together. If the line in l_{XOR} includes the lines l_C and l_D, l_{XOR} will be kept; otherwise, it will not be kept. We use the following equation to find the area that needs to be inpainted.

$$l'_{XOR} = \begin{cases} 1, & \text{if } l_C \subset l_{XOR} \text{ and } l_D \subset l_{XOR} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

Fig. 4 shows an example of l'_{XOR}, where we can see that unnecessary textures in the color image have been removed successfully. The remaining white parts are the unmatched locations of edges between the two images, which are the areas that need to be inpainted. In Fig. 4, the area pointed by a red arrow is an example of obvious unmatched locations.

B. Selecting Inpainting Areas

Inpainting is conducted based on l'_{XOR}. In this paper, the direction for inpainting is determined based on whether pixels in l'_{XOR} are on the interior or exterior of the object in l'_{XOR} image. Therefore, the objects in the image must be extracted first. Furthermore, to inpaint l'_{XOR} better, two cases must be excluded before inpainting is conducted: (1) areas

that are too small, (2) the remaining l'_{XOR} does not form a complete closed area. The whole process is shown in Fig. 5.

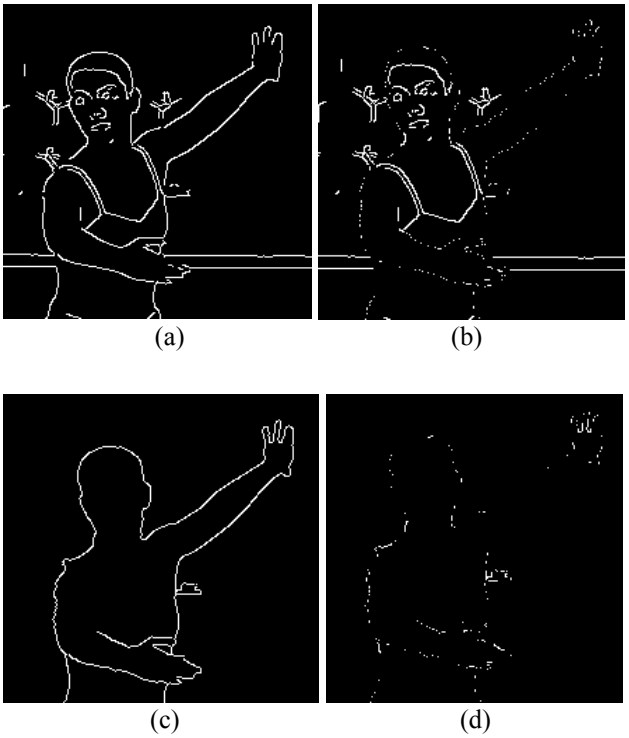


Fig. 3 A demonstration of edge removal. (a) $\boldsymbol{I_C}$: edge information of original color image; (b) $\boldsymbol{I'_C}$: edge information remaining after edge removal (c) $\boldsymbol{I_D}$: edge information of original depth image; (d) $\boldsymbol{I'_D}$: edge information remaining after edge removal.

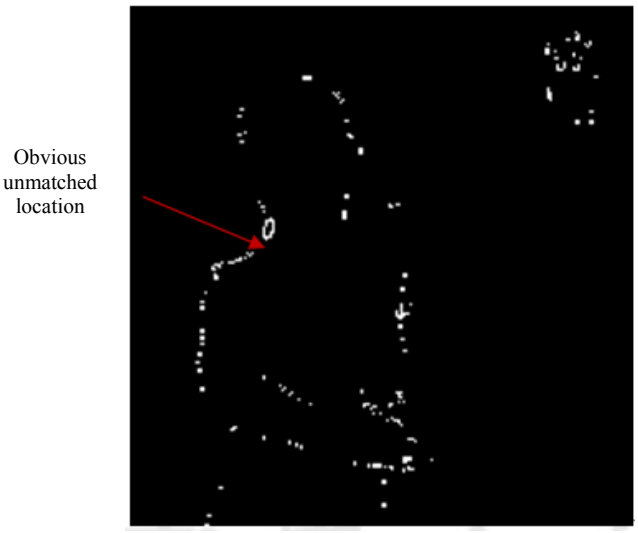


Fig. 4 Edge information remaining after using Eq. (4).

For object extraction, after edge detection, we identify the objects that belong to the same edge line. Here, we utilize I_D to extract objects from the image because we can extract a foreground object from a depth image more completely (with less broken or discontinuous edges). However, if any object in a depth image has discontinuous edges, then it will be treated directly as two separate objects without further processing. In Fig. 6(a), an object is divided into two line segments due to discontinuous edges. In this case, it will be treated as two separate objects, as shown in Fig. 6(b).

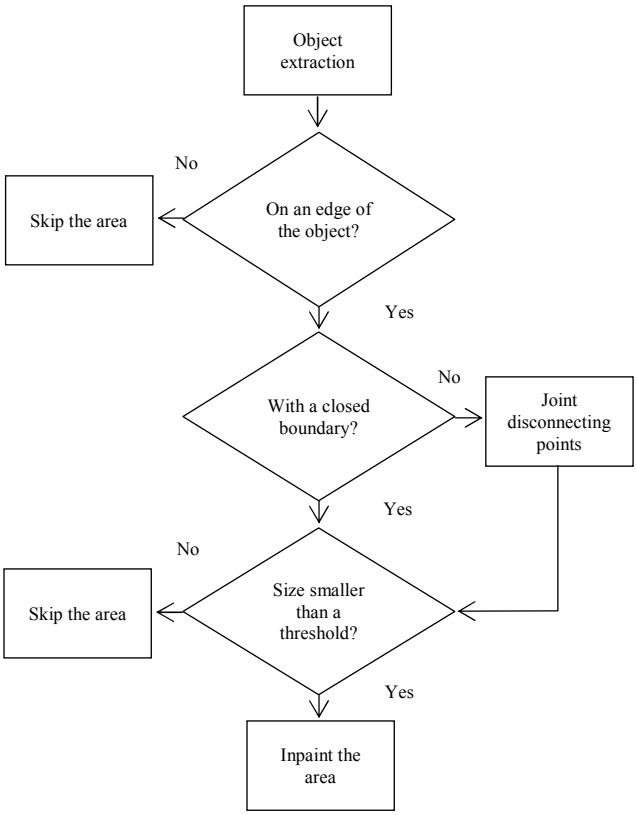


Fig. 5 A flowchart of selecting inpainting area.

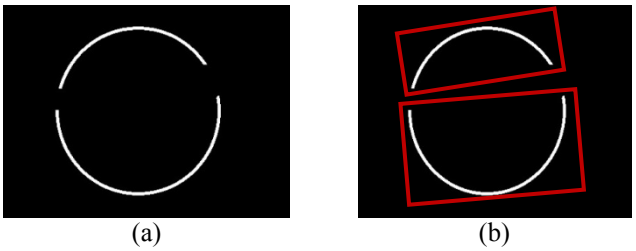


Fig. 6 Object extraction in the case of discontinuous edges. (a) an object that is divided into two line segments due to discontinuous edges; (b) the object in (a) is treated as two separate objects.

First, we find the common edges I_{AND} shared by I_C and I_D.

$$I_C * I_D = I_{AND} \tag{5}$$

Fig. 7 shows an example of common edges obtained from Eq. (5).

Fig. 7 Common edges $\boldsymbol{I_{AND}}$ shared by $\boldsymbol{I_C}$ and $\boldsymbol{I_D}$

After finding the common edges, we extract the objects in I_D (as shown in Fig. 8(a)), and each object is bounded by a rectangular frame. The extracted object frames are applied to the corresponding locations in I_{AND}. For example, if the location coordinates of object frames in I_D are (10 , 20), (30 , 20), (10 , 40), and (30 , 40), then the object frames at the same locations in I_C and I_{AND} are used to find the objects in I_C and I_{AND}.

Next, count the number of edge pixels in each object. Denote this number for an object in I_{AND} and I_D as $Count_{AND}$ and $Count_D$, respectively. Consider a similarity measurement as follows:

$$S = \frac{Count_{AND}}{Count_D} \tag{6}$$

An object with very low similarity measurement is removed with Eq. (7) because it may be textures inside the object.

$$I'_{AND} = \begin{cases} O \text{ in } I_{AND}, & \text{if } S > \text{threshold} \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

where O is the area deemed as object after calculation. Fig. 8(b) gives the image of I'_{AND}. Compared to Fig. 8 (a), some selected object areas (in red) are removed. This step can remove some areas that are textures instead of object edges. Finally, the location of object O in I'_{AND} is applied to I_C to identify the corresponding object in I_C. The O in I_C is the object that we want to inpaint. Incidentally, the rectangular frames in Fig. 8 are used merely to demonstrate the objects selected. In fact, the object contours rather than the rectangular frames are the real O of interest.

After object selection, the areas required to be inpainted are dealt with in three parts: (1) areas not on the object edge must be removed, (2) areas that are too small shall be removed, and (3) the edges of an open area to be inpainted are connected to form a closed area. The whole process is shown in Fig. 9. The blue line in Fig. 9(a) denotes the boundary of an object O; The red line indicates the case when a closed area is on the object boundary; The green ellipse is the area outside of object boundary; The purple line represents the case of an open area; The yellow line shows the case where the area is too small. Firstly, we need to determine if the areas to be inpainted belong to the areas of O and are on the boundary of O. For example, since the green ellipse is not on the boundary of O, it will be removed (Fig. 9(b)). Next, determine if there is any open area, such as the purple lines shown in Fig. 9(a). Let the two disconnected end points of the two purple lines be $A(x_1, y_1)$ and $B(x_2, y_2)$, respectively. Then the distance between A and B can be calculated by:

$$|AB| = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{8}$$

(a) (b)

Fig. 8 Correspondent object selection. (a) Objects selected in depth image; (b) I'_{AND} image.

Find the closest open lines. If the distance of the two closest end points is smaller than a threshold, i.e. $|AB| < T_d$, then the two points are connected with a straight line, where T_d is a user-specified distance parameter. If the distance is larger than T_d, remove those two lines without trying to connect them. Fig. 9(c) shows a schematic that an open area becomes closed. When all areas are closed areas, remove smaller areas because inpainting on areas that are too small increase the computational cost and get no obvious benefit. In fact, according to our experiment, inpainting the area with size smaller than 70 pixels gives no noticeable visual improvement. After removing the areas that are too small, the remaining areas are the areas that require inpainting, as shown in Fig. 9(d). Fig. 10 shows a typical processing result. Fig. 10(a) shows all areas of interest and Fig. 10(b) is the remaining areas after the processing. Obviously, some areas have been removed. Here, an object as well as its associated inpainting areas are obtained and expressed as $O = \{b_1, b_2, \cdots, b_n\}$, where b_r represents the rth area in the object that requires inpainting.

C. *Dual Edge Inpainting*

In area inpainting, the proposed approach, which is different from other inpainting methods, uses the edges in a depth image and the edges in the corresponding color image to conduct directional inpainting. If an inpainting area is inside the object in the color image, the inpainting direction is from the depth image side to the color image side; otherwise, the direction is from the color image side to the depth image side. After the inpainting direction is determined, the pixel points on an object edge are chosen to conduct inpainting first. The inpainting is performed with the fast edge pixel

interpolation method described in Part B of Section II. The inpainting algorithm is given as follows:

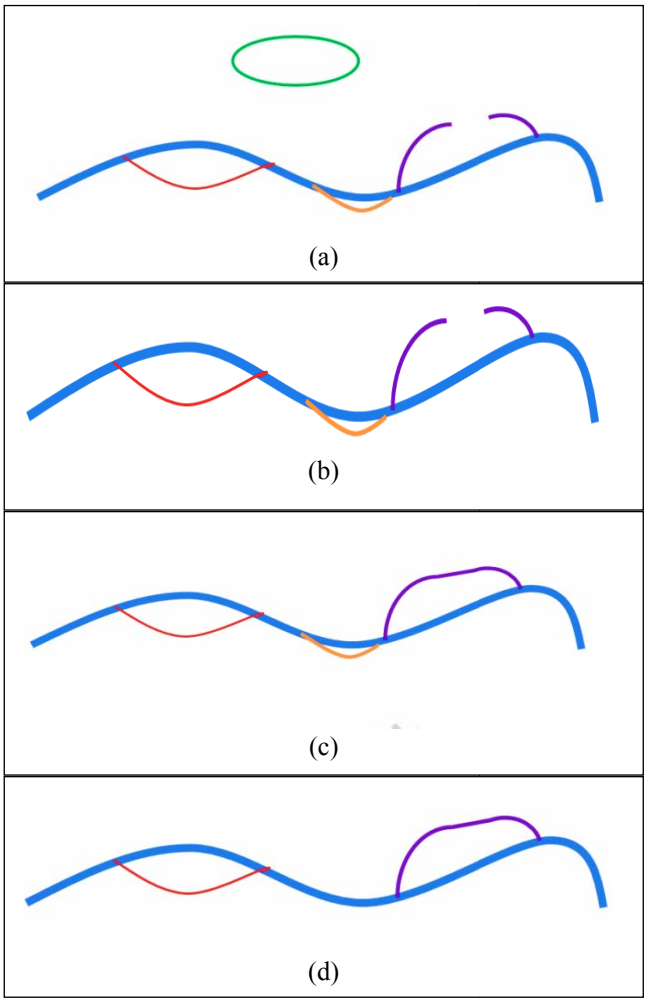


Fig. 9 Selecting and dealing with inpainted area. The red curve denotes a closed area on object boundary. The green ellipse indicates an area not on object boundary. The orange curve represents an area that is considered too small. The purple one represents a disconnected area. (a) All types of areas that may occur; (b) Remove areas not on object boundary; (c) Connect edges of open areas; (d) After the smaller area is removed, the areas which require to be inpainted are kept.

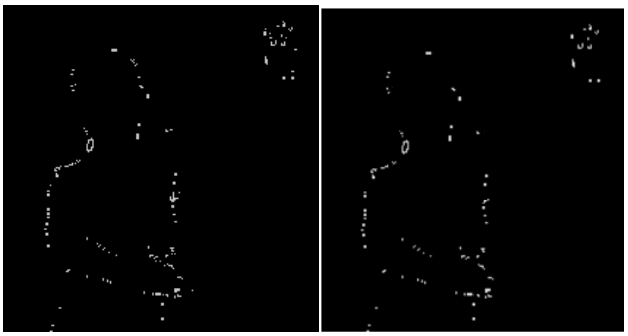


Fig. 10 A typical result of selecting inpainting areas. (a) the image before the process of inpainting area selection; (b) the image after the process of inpainting area selection.

Step0. Given an object $O = \{b_1, b_2, \cdots, b_n\}$, where b_r represents the rth area in the object that requires inpainting. I_C, I_D, l_C and l_D are defined in Part A of Section III.

Step1. Determine inpainting direction for each area of interest
If b_r is inside O, direction of inpainting is $I_D \rightarrow I_C$.
If b_r is outside O, direction of inpainting is $I_C \rightarrow I_D$.

Step2. $I_D \rightarrow I_C$ inpainting: Inpaint points on I_D first.

Given $f(i,j) \in b_r$ and $count = 0$;

for $f(i+k, j+k)$, $k = \{-1, 0, 1\}$;

if $f(i+k, j+k) \in l_D$ $count = count+1$;

if $count \geq 3$ *then* $f(i,j) \in l_D$, perform edge area pixel interpolation;

else $f(i,j) = \frac{1}{count} \sum_{k=-1}^{1} f(i+k, j+k)$, where $f(i+k, j+k') \in l_D$;

if $f(i,j) \in l_C$ Stop;

The inpainting for $I_C \rightarrow I_D$ is the same as that for $I_D \rightarrow I_C$ except that I_C and I_D as well as l_C and l_D are exchanged in Step2. Fig. 11 shows the inpainting result with the proposed dual edge inpainting method. Fig. 11(a) and 11(c) are the original images that have not been inpainted, while Fig. 11(b) and 11(d) are the images that have been inpainted, respectively. Firstly, Fig. 11(a) and 11(b) are an example of inward inpainting. We can see that at the dancer's shoulder in Fig. 11(a) (in red ellipse), the original depth image does not match the original color image. After performing the dual edge inpainting, the extruded part on the shoulder is contracted inward so that the shoulder portion of the depth image can match the color image better. Fig. 11(c) and 11(d) are an example of outward inpainting. The areas inside red ellipses are the 'holes' after comparing with the depth image. After inpainting, the holes are filled quite successfully. Note that Fig. 11(a) and 11(c) are sub-images within the same image. Therefore, inward inpainting and outward inpainting can be conducted simultaneously for different inpainting areas in the same image.

IV. Experimental Results and Discussion

This experiment is aimed at the evaluation of edge consistency between depth images and the corresponding color images. The video data for the experiment are the sequences of Ballet and Break-dancer provided by the Microsoft. The image size is 1024×768. Both sequences have 100 frames each. The software used in the experiment is Microsoft Visual Studio 2008 C++ and JMVC (Joint Multiview Video Coding), as well as OpenCV2.3. The experiment consists of two main parts: (1) comparison of edge consistency and (2) compression effect with different QP (quantization parameter) values. In addition to the use of traditional de-blocking filters (hereafter referred as Method 1) for comparison, we also make comparison with Trilateral Filter + SD Mode (hereafter referred as Method 2) in paper [14]. The method of Trilateral Filter + SD Mode is an

algorithm used after de-blocking. Therefore, the proposed method is used after deblocking as well (Fig. 12). More details on the first and the second part of the experiment will be given in Part A and Part B of this section, respectively.

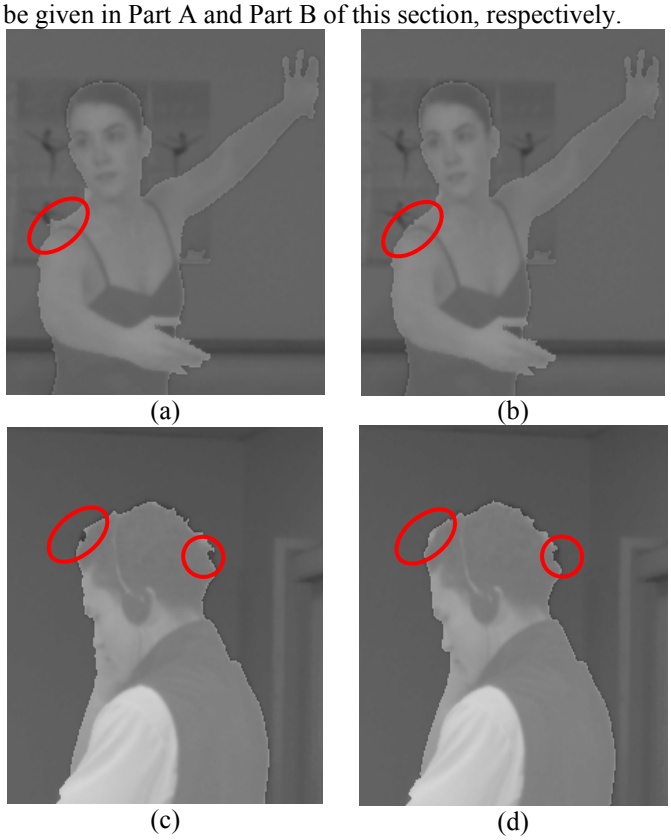


Fig. 11 An example of using dual edge inpainting method. (a) before inward inpainting; (b) after inward inpainting ; (c) before outward inpainting; (d) after outward inpainting.

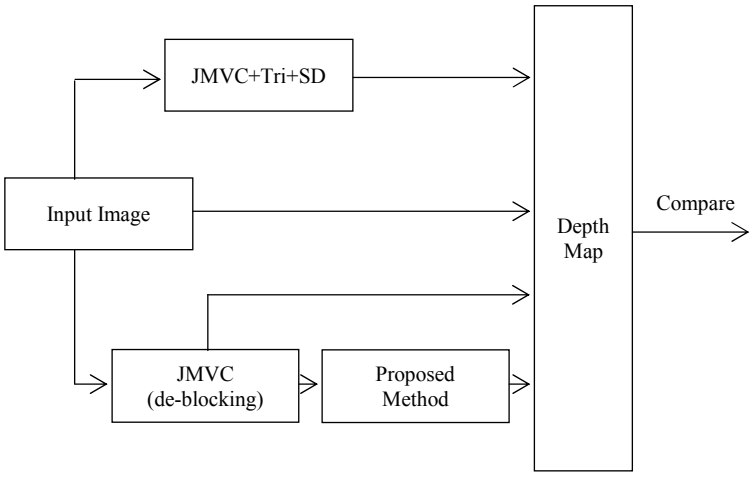


Fig. 12 A schematic showing the comparison study in the experiment

A. Comparison of Edge Consistency

According to the setting of Fig. 12, various depth images are generated from the original depth images. Then edge detection is performed on all the depth images. For each image frame, say frame *i*, the number of common edges (detected by AND operation) between the derived and the original depth images and the number of edges in the original depth image are recorded for the calculation of a similarity measure as follows:

$$S_i = \frac{Number\ of\ edges\ after\ AND\ computing}{Number\ of\ edge\ of\ original\ depth\ image} \times 100\% \quad (9)$$

The average performance for all images in a video sequence is given by

$$Avg = \frac{1}{n}\sum_{i=1}^{n} S_i \quad (10)$$

where n is the total number of frames in the sequence considered here, which is n = 100. Table I gives the average result to reflect the performance of edge consistency among the three methods considered here. From Table I, we can see that Method 2 preserves more edges in original depth image than that of Method 1. In addition, serving as a post-processing technique, the proposed method can significantly enhance the edge similarity metric obtained from Method 1. Table II shows PSNR performance for depth images. The proposed method enhances the PSNR performance of Method 1 by more than 1 dB. The PSNR performance for the overall method consisting of Method 1 and the proposed method is comparable to that of Method 2.

TABLE I
EDGE SIMILARITY MEASUREMENT FOR DEPTH IMAGES

Method / Video Sequence	Method 1	Method 2	Method 1+ Proposed Method
Ballet	67.3%	86.4%	86.5%
Break-dancers	55.6%	76.2%	77.1%

TABLE II
AVERAGE PSNR PERFORMANCE FOR DEPTH IMAGES (UNIT: dB)

Method / Video Sequence	Method 1	Method 2	Method 1+ Proposed Method
Ballet	36.1	37.6	37.4
Break-dancers	35.5	37.9	37.7

Following the same manner as before, we first derive the depth images with various methods. Then we perform edge detection on the derived depth images and the corresponding color images. Again, Eq. (9) and Eq. (10) are used as a measure of edge consistency but the 'original depth image' in the denominator of Eq. (9) is replaced by the 'original color image.' The performance results are given in Table III, where four methods are considered, namely, Edges in Depth Image, Method 1, Method 2, and Method 1 + Proposed method. From Table III, we see that the proposed method can enhance the edge similarity metric for Method 1 significantly, it

outperforms Method 2 by about 4 to 5%, and its edge consistency is almost as good as the original depth image.

B. Compression Effect with Different QP Values

Given several QP values in JMVC, we investigate the PSNR performance of the three methods for depth images. A QP value in JMVC represents the degree of image compression; the higher QP value we use, the higher compression ratio will be expected. Similarly, a lower QP value usually results in lower compression ratio. The QP step is set to 6 units here because human being can directly sense the quality difference of decompressed images when the variation of QP is in the step of 6 scales according to the investigation in [16]. Table IV and Table V give the results for the two video sequences, respectively. After each depth image is processed by the three methods, the PSNR values are calculated, and the average PSNR values for the entire sequence is taken to produce the results given in Table IV and Table V. We learn from these two tables that both Method 2 and the proposed method perform better than Method 1 by 0.5 to 1.5 dB. In addition, the proposed method has a comparable performance to Method 2 when QP = 12 and QP = 18. For QP = 24 and QP = 30, the proposed method has 0.8 dB advantage over Method 2. However, when the QP value reaches 36, the performance of the proposed method drops by about 4 dB, and there is no obvious improvement over Method 1 in this case. The reason for a sudden performance drop when QP = 36 is the significant drop of edge information contained in the color image due to the high degree of compression. Since inpainting is done after decompression, the proposed method may not work well when compression on color images brings too much distortion and destroys useful edge information. As compared to the proposed method, Method 2 is relatively steadier when the QP value increases from 30 to 36. However, the authors of [16] also pointed out that when a QP value higher than 30 is chosen, the variation of image quality after decompression is too large to be acceptable or the image quality may be too poor to have any practical interest.

V. CONCLUSIONS

This paper proposes a post-processing method to inpaint depth images. The difference between this paper and the precedents is that most inpainting methods inpaint depth images directly without considering the correlation between a depth image and its corresponding color image. However, in [14], with Trilateral filter + SD Mode, the correlation of edges between color images and depth images are calculated. High correlated edges are heavily protected during image transmission to increase the completeness of depth image. The common part of this study and the study in [14] is to use the edge information from both color image and depth image. The difference between these two studies is that the proposed method operates as a post-processing tool at decoder side, while the method proposed in [14] is operated at encoder side. In addition, for the unmatched locations with color images, a dual edge inpainting technique is proposed to directionally inpaint the depth image that has mismatched locations with the corresponding color image. Both the method of Trilateral filter + SD Mode and the proposed method can enhance the quality of depth image. Our experimental results show that the proposed method and the method of Trilateral filter + SD Mode have a comparable performance when the QP values from 12 to18 are set for JMVC. When the QP values are from 18 to 24, the proposed method has better performance. Since the proposed approach is basically a post-processing technique after an image is received and decoded, it can be applied to the images transmitted via other 2D + Depth transmission mechanisms.

According to the suggestion from the reviewers of this paper, one important future work is to conduct a subjective experiment, where some viewers are invited to watch actual 3D videos (not just the depth maps) in order to verify the performance of the proposed method. An in-depth comparison study with other relevant approaches was also recommended. These comments are appreciated and they would become the guidance of our future research.

TABLE III
SIMILARITY MEASURE WITH EDGES IN COLOR IMAGE

Method / Video Sequence	Edges in Depth Image	Method 1	Method 2	Method 1+ Proposed Method
Ballet	43.2%	27.6%	36.5%	41.7%
Break-dancers	36.1%	25.4%	32.3%	36.2%

TABLE IV
AVERAGE PSNR PERFORMANCE FOR DEPTH IMAGES (UNIT: dB): BALLET SEQUENCE

Method / QP	Method 1	Method 2	Method 1+ Proposed Method
12	37.3	39.2	39.3
18	36.1	37.9	37.7
24	35.8	36.5	37.4
30	34.3	36.1	36.9
36	32.3	34.9	32.6

TABLE V
AVERAGE PSNR PERFORMANCE FOR DEPTH IMAGES (UNIT: dB): BREAK-DANCERS SEQUENCE

Method / QP	Method 1	Method 2	Method 1+ Proposed Method
12	38.0	39.3	39.3
18	36.5	37.6	37.4
24	36.0	36.4	37.4
30	34.1	35.3	36.4
36	32.3	33.5	32.7

REFERENCES

[1] H. J. Feng, "PK of Eye Revolutionary Display Technology," Wealth Invest Weekly, no. 1687, 2012.

[2] H. Y. Tsai "3DTV popular? —2010, Start 3DTV Testing," Strategic R & D Department, Taiwan Public Television, 2010.

[3] J. F. Chung "Is it? It Is Not Dream of 3D Wireless Transmission of High-Quality," CTimes News, 2010.

[4] Joy, WiGig (Wireless Gigabit Alliance), Developing a New Generation of Wireless LAN Standards, Science & Technology Policy Research and Information Center, 2009. http://cdnet.stpi.org.tw/techroom/analysis/2009/pat_09_A016.htm

[5] WiGig White Paper Define the Future of Multi-Gigabit Wireless Communications, July 2010. http://wirelessgigabitalliance.org/specifications/

[6] T. Okino, H. Murata, K. Taima, T. Iinuma, and K. Oketani, "New Television with 2D/3D Image Conversion Technologies," *Proc. of SPIE*, vol. 2653, pp. 96-103, 1996.

[7] L. Zhang and W. J. Tam, "Stereoscopic Image Generation Based on Depth Images for 3D TV," *IEEE Trans. on Broadcasting*, vol. 51, Issue 2, pp. 191-199, June 2005.

[8] E. Ekmekcioglu, M. Mrak, S. T. Worrall and A. M. Kondoz, "Edge Adaptive Upsampling of Depth Map Videos for Enhanced Free-viewpoint Video Quality," *IET Trans. on Electronics Letters*, vol. 56, no. 7, pp. 353-354, March 2009.

[9] L. M. Po, S. Zhang, X. Xu, and Y. Zhu, "A New Multidirectional Extrapolation Hole-Filling Method for Depth-Image-Based Rendering," *Proc. of IEEE Conf. on Image Processing*, pp. 11-14, Sept. 2011.

[10] H. Y. Chai, Directional Hole-Filling Method for 3D View Generator, Master Thesis, Department of Computer Science and Information Engineering, Tamkang University, 2012.

[11] Y. C. Liu, Depth Map Modify System of Three-Dimensional Stereo Image, Master Thesis, Graduate Institute of Computer and Communication Engineering, National Taipei University of Technology, 2009.

[12] K. J. Yoon and I. S. Kweon, "Adaptive Support-Weight Approach for Correspondence Search," *IEEE Trans. on Pattern Anal. Mach. Intell.*, vol. 28, pp. 650-656, 2006.

[13] M. Mueller, F. Zilly, and P. Kauff, "Adaptive Cross-Trilateral Median Filter," *Proc. of IEEE Conf. on The True Vision - Capture, Transmission and Display of 3D Video*, pp. 1-4, June 2010.

[14] S. Liu, P. L. Lai, D. Tian and C. W. Chen, "New Depth Coding Techniques with Utilization of Corresponding Video," *IEEE Trans. on Broadcasting*, vol. 57, no. 2, pp. 551- 561, June 2011.

[15] J. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. on Pattern Anal. Mach. Intell.*, vol. 6, pp. 679-698, 1986.

[16] Y. C. Li, H.264/SVC Rate Allocation based on Graceful Degradation of Subjective Quality in Frame Rate Switching, Master Thesis, Department of Communication Engineering, National Central University, 2010.