Journal of Signal and Information Processing, **2013, 4, 327-335**
http://dx.doi.org/10.4236/jsip.2013.43041 Published Online August 2013 (http://www.scirp.org/journal/jsip)

Depth Based View Synthesis Using Graph Cuts for 3DTV

Anh Tu Tran, Koichi Harada

Graduate School of Engineering, Hiroshima University, Higashihiroshima, Japan.
Email: d103714@hiroshima-u.ac.jp, hrd@hiroshima-u.ac.jp

Received April 12th, 2013; revised May 12th, 2013; accepted June 12th, 2013

ABSTRACT

In three-dimensional television (3DTV), an interactive free viewpoint selection application has received more attention so far. This paper presents a novel method that synthesizes a free-viewpoint based on multiple textures and depth maps in multi-view camera configuration. This method solves the cracks and holes problem due to sampling rate by performing an inverse warping to retrieve texture images. This step allows a simple and accurate re-sampling of synthetic pixels. To enforce the spatial consistency of color and remove the pixels wrapped incorrectly because of inaccuracy depth maps, we propose some processing steps. The warped depth and warped texture images are used to classify pixels as stable, unstable and disoccluded pixels. The stable pixels are used to create an initial new view by weighted interpolation. To refine the new view, Graph cuts are used to select the best candidates for each unstable pixel. Finally, the remaining disoccluded regions are filled by our inpainting method based on depth information and texture neighboring pixel values. Our experiment on several multi-view data sets is encouraging in both subjective and objective results. Furthermore, our proposal can flexibly use more than two views in multi-view system to create a new view with higher quality.

Keywords: View Synthesis; Depth Image Based Rendering (DIBR); Free-Viewpoint TV; Graph Cuts

1. Introduction

Recently, 3D-TV application and system are rapidly growing. With the growing capability of capturing devices, multi-view capture system with dense or sparse camera array can be built with ease, free-viewpoint television (FTV) [1] system has attracted increasing attention. In FTV system, users can freely select the viewpoint of any dynamic real world to see. The chosen free-viewpoint cannot only be selected from available multi-view camera views, but also from any viewpoint between these cameras. This system requires a smart synthetic algorithm that allows free-viewpoint view rendering. To render a high quality image at an arbitrary viewpoint, one has to manage three main challenges as pointed out in [2]. First, empty pixels and holes due to sampling of the reference image have to be closed. Secondly, pixels at borders of high discontinuities cause contour artifacts. The third challenge involves inpainting disocclusions that remain after blending the projected images (these are invisible from any of the surrounding cameras). In [3] it is shown that one can obtain an improved rendering quality by using the geometry of the scene. When using depth information, a well-known technique for rendering is called Depth Image Based Rendering (DIBR), which involves the 3D-projection or 2D-warping from a viewpoint into another view.

In this paragraph, we describe briefly some recent researches on free-viewpoint DIBR algorithm. In [2], the author has developed a free-viewpoint rendering algorithm which is based on layered representation. For texture mapping, 3D meshes are created and the rendering is implemented on a Graphics Processing Unit (GPU). Although the results look good, the method is complex and requires a considerable amount of pre- and post-processing operations. This work is extended in [4] where the depth map is decomposed into three layers and these layers are warped separately. The warp results are obtained for each layer and merged. To deal with artifacts, they have introduced three post-processing algorithms. In [5], a new viewpoint is rendered by some steps. First, the depth maps of the reference cameras are warped to the new viewpoint. Then the empty pixels are filled with a median filter. Afterwards, the depth maps are processed with a bilateral filter. Then, the textures are retrieved by performing an inverse warping from the projected depth maps back to the reference cameras. Ghost contours are removed by dilating the disocclusions. Finally, the tex-

ture images are blended and the remaining disocclusions are inpainted using the method proposed by Telea [6]. Although, the results look good, this method is remaining some issues such as not removing all holes by median filter, assigning a none-zero value for some pixels in disocclusion regions. This work is improving in [7] by introducing three enhancing techniques. First, re-sampling artifacts are filled in by a combination of median filtering and inverse warping. Second, contour artifacts are processed while omitting warping of edges at high discontinuities. Third, disocclusion regions are inpainted with depth information. The quality of this method is higher than the work in [5], but still having disadvantages. For example, they have to define the label of pixel at high discontinuities. The color consistency during blending is not verified to avoid jagged edges at straight line after blending. The work in [8] combines depth based hole filling and inpainting to restore the disoccluded pixels more accurately compared to inpainting method without using depth information. This method produces a notable blur and can be computationally inefficient when disoccluded region is larger in the new view.

In this paper, we introduce a new free-viewpoint rendering algorithm from multiple color and depth images. First, the depth maps for the virtual views are created by warping the depth maps of reference cameras. We process the wrapped depth maps with median filter. Depth maps consist of smooth regions with sharp edges, so filtering with a median will not degrade the quality. Then, the textures are retrieved by performing an inverse warping from the warped depth maps to the reference cameras. This allows a simple and accurate resampling of synthetic pixels. After that, all warped depth and warped texture images are used to classify pixels as stable, unstable and disoccluded regions. An initial virtual view is created based on weighted interpolation of stable pixels. To refine the synthetic view, the best candidates for unstable pixels are optimally selected by Graph cuts. By defining the types of pixels and using Graph cuts, the color is consistent and the incorrectly wrapped pixels because of inaccuracy depth maps are removed in the refined view. The remaining disoccluded pixels are inpainted by using depth and texture neighboring pixel values. Considering depth information for inpainting, blurring between foreground and background textures is reduced.

The rest of this paper is organized as follows: Section 2 presents the proposed view synthesis algorithm. Section 3 shows experimental results; and, finally, Section 4 concludes this paper.

2. Proposed Synthesis Method

Our proposal is shown in **Figure 1** and it consists of six steps. These steps are explained below.

2.1. 3D Warping the Depth Maps

3D warping enables to synthesize a new view from the reference view as following.

Let $P_w = [X_w, Y_w, Z_w, 1]^{\mathrm{T}}$ be the world point; $p_1 = [u_1, v_1, 1]^{\mathrm{T}}$ and $p_2 = [u_2, v_2, 1]^{\mathrm{T}}$ be its projection onto reference and synthetic image planes, respectively. P_w, p_1 and p_2 are related by the camera perspective projection (1) and (2).

$$\lambda_1 p_1 = K_1 [R_1; -R_1 C_1] P_w \tag{1}$$

$$\lambda_2 p_2 = K_2 [R_2; -R_2 C_2] P_w \tag{2}$$

where, K_i is a 3×3 upper triangular matrix representing the inner structure of the camera i and is called the intrinsic matrix. The 3×3 orthogonal matrix R_i represents the orientation and $3-$vector C_i represents the position. The matrix $[R_i; -R_i C_i]$ is called the extrinsic matrix and it indicates the relationship between world coordinates and the camera coordinates.

Rearranging (1) we can derive 3D coordinate of the scene point P_w:

$$(X_w, Y_w, Z_w)^{\mathrm{T}} = (K_1 R_1)^{-1} \cdot (\lambda_1 p_1 + K_1 R_1 C_1) \tag{3}$$

Substituting (3) into (2) we obtain the synthetic pixel position p_2:

$$\lambda_2 p_2 = K_2 R_2 (K_1 R_1)^{-1} \cdot (\lambda_1 p_1 + K_1 R_1 C_1) - K_2 R_2 C_2 \tag{4}$$

Assuming that the world coordinate system is the same as the reference camera coordinate system and looks at along $Z-$direction, *i.e.*, $C_1 = (0,0,0)$, $R_1 = I_{3x3}$ and $\lambda_1 = Z_w$, Equation (4) can rewrite as following:

$$\lambda_2 p_2 = K_2 R_2 K_1^{-1} \cdot Z_w p_1 - K_2 R_2 C \tag{5}$$

where, Z_w is defined by the pixel value at coordinate point p_1 in the reference image.

Applying (5) for a point $p_1 = [u_1, v_1, 1]^{\mathrm{T}}$ from the reference image we can calculate a point p_2 on the synthesis image. The problem that several points can be projected to the same point in virtual image is solved by using simple $z-buffering$ technique. Another issue of this process is that a pixel p_1 of reference view is not usually projected on to a point p_2 at integer pixel position. To obtain an integer pixel position, we map the sub-pixel p_2 to the nearest integer pixel $\hat{p}_2$ as follows equation:

$$\hat{p}_2 = (\hat{x}_2, \hat{x}_2, 1) = (\lfloor x_2 + 0.5 \rfloor, \lfloor y_2 + 0.5 \rfloor, 1) \tag{6}$$

In our method, only depth maps of reference cameras are projected to virtual image plane. The warping is specified by:

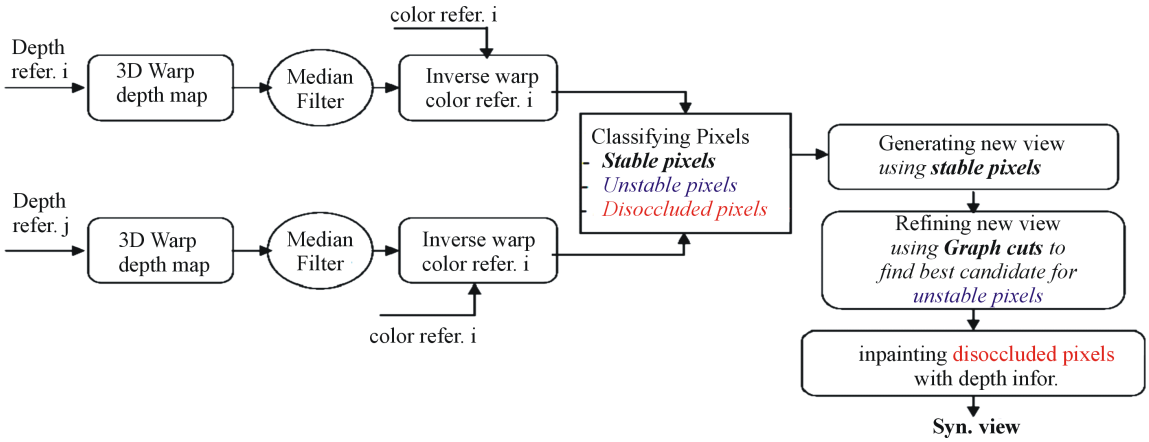


Figure 1. Proposed new view synthesis algorithm.

$$\left[Z_{syn}, \hat{p}_2\right] = 3D\_\text{warp}\left(Z_{ref}, p_1\right), \quad (7)$$

where, Z_{ref} is depth map of a reference camera, $3D\_\text{Warp}$ is warping operation as above describing. The projected depth maps from two reference cameras for an arbitrary scene are shown in **Figure 2**.

2.2. Median Filter the Warped Depth Map

In this step, we consider the blank points that appeared in projected depth map. The reasons for the appearance of these blank points are round off errors of the image coordinate by (6) and depth discontinuities. It can cause one pixel wide blank region to appear. This blank region can be filled by median filter with a window of 3×3 pixels. Depth maps consist of smooth regions with sharp edges, so filtering with a median will not degrade the quality.

This step can describe as:

$$Z_{syn\_filtered} = Median(Z_{syn}), \quad (8)$$

where, $Median$ is a median filter with a window 3×3 pixels, $Z_{syn\_filtered}$ is output of median filter. The image in **Figure 2** can be processed by using median filter to obtained images in **Figure 3**.

2.3. Retrieve Texture Image by Inverse Warping

In this step, the textures are retrieved by performing inverse warping from filtered projected depth maps back to the reference cameras.

For each pixel p_2 of the filtered projected depth image, a $3D$ world point $P_{2w} = \left(X_{2w}, Y_{2w}, Z_{2w}\right)$ is calculated based on (3). Z_{2w} is defined by the depth value at coordinate p_2 in the filtered projected depth image. Then, the calculated $3D$ point P_{2w} is projected onto respective reference textures image by employing (5), such that color of the synthetic destination pixel p_2 is interpolated from the surrounding pixel p_1 in the reference color image. **Figure 4** illustrates the image rendering process using inverse warping.

This step can be specified by:

$$\left[I_{syn}, p_2\right] = 3D\_\text{warp}^{-1}\left(Z_{syn-\text{filtered}}, p_2\right), \quad (9)$$

The advantage of an inverse warping operation is that all pixels of the destination image are correctly defined and the color disoccluded pixels can be inferred by back

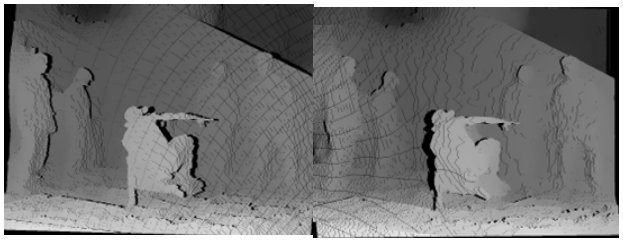


Figure 2. The projected depth maps from two reference cameras (from the left side and from the right side).

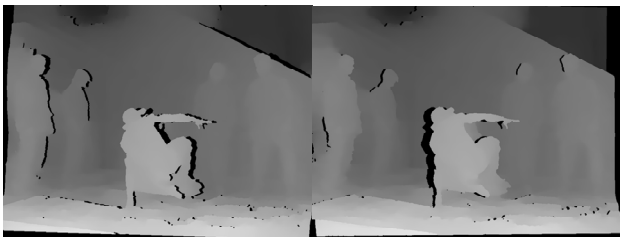


Figure 3. Median filter depth maps.

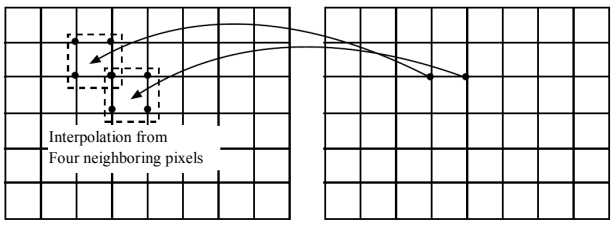


Figure 4. Image synthesis process using inverse warping.

projected $3D$ point P_{2w} onto multiple source image planes, covering all regions of video scene.

Figure 5 shows the retrieved color images by inverse warping using depth maps in **Figure 3**.

2.4. Pixel Classification and Initial New View Creation

Formally, suppose that we have a set of N texture images $I=\{I_1, I_2, \cdots, I_N\}$ and N depth images $Z=\{Z_1, Z_2, \cdots, Z_N\}$. Let $I_m(p)$ and $Z_m(p)$ be the color and depth value at position of $m-th$ image.

In this step, we describe the type of pixels in the synthetic view. We go through each pixel $p \in P$ of all N input images and classify as stable, unstable and disoccluded pixels. To detect the types of pixel, we set the thresholds (depth threshold t_Z and color threshold t_C) and examine the color and depth values for pixel $p \in P$. For each color channel, the color threshold t_C is set to be 15 in our case. Depth threshold is the brightness in the depth map. In our experiments, t_Z is set to 5 for the 8 bits depth quantization.

A pixel is classified as:

If the depth value of a pixel $p \in P$ at all N input depth images is less than depth threshold t_Z, we classify the pixel p as the *disoccluded pixel*. The color and depth values of the pixel p at synthetic view are set temporally to zero.

$$I_{\text{new}}(p)=0, Z_{\text{new}}(p)=0, \text{ if } Z_k(p) \le t_Z, \ \forall k=1,2,\cdots,N. \tag{10}$$

If the depth value of a pixel $p \in P$ at only one input image is higher than the depth threshold t_Z and at all remaining $(N-1)$ images is less than t_Z, we classify the pixel p as the stable pixel. This is case the pixel p is visible in only one view. The values of the pixel p at synthetic view are just copied from the values of the pixel p in the visible view.

$$I_{new}(p)=I_k(p), Z_{new}(p)=Z_k(p), \quad \text{if } Z_k(p)>t_Z, Z_m(p)\le t_Z, \forall m=1,2,\cdots,N, m\ne k. \tag{11}$$

If the depth value of a pixel $p \in P$ is higher than the depth threshold t_Z in more than one view, we examine both the color and depth values of the pixel p to detect the types of pixel.

Figure 5. Obtained color images by inverse warping.

First step, for each view k, $k=1,2,\cdots,N$, we examine pixel p. If the depth value of the pixel p is higher than the depth threshold t_Z, then we check other views j, $j=1,2,\cdots,N$, $j \ne k$. If the view j has both a depth value of the pixel p higher than the depth threshold t_Z and has color similarity at p of view j and k, $I_j(p)$ and $I_k(p)$ are called consistent color (the color similarity at pixel p of two input images j and k is defined based on the absolute color differences between $I_j(p)$ and $I_k(p)$ of R, G and B channels, $|I_j(p)-I_k(p)|<t_C$). We count the total number of view j, $j=1,2,\cdots,N$ having the consistent color with view k $(k=1,2,\cdots,N, j\ne k)$ at pixel p. Assuming that for each view $k=1,2,\cdots,N$, this total number is S_k.

Second step, we find the biggest number of S_k, assuming that the biggest number is M.

If $M \ge \lfloor N/2+0.5 \rfloor$, we classify the pixel p as the stable pixel. Otherwise, the pixel p is classified as the unstable pixel. The value of unstable pixel can set to be -1 so that they can be easily identified.

The color and depth values of stable pixel p at synthetic view are rendered by blending M pixels as following weighted interpolation:

$$\begin{aligned} I_{\text{new}}(p) &= \left(\sum_{i=1}^{M} w_i * I_i(p)\right) \Big/ \left(\sum_{i=1}^{M} w_i * z_{\text{new}}(p)\right) \\ &= \left(\sum_{i=1}^{M} w_i * z_i(p)\right) \Big/ \sum_{i=1}^{M} w_i, \end{aligned} \tag{12}$$

where, w_i is the weight factor assigned to view i, $I_i(p)$ and $Z_i(p)$ are color value and depth value of pixel p at view i. The weight assigned to each view should reflect its proximity with the view being synthesized. The views that are closer to the synthetic view should have a bigger weight. In general, case, the weight w_i can be set based on baseline spacing. However, for more precise weighting, we use the angle distance determined by the point in $3D$ and camera positions as shown in **Figure 6**. The weight factor w_i is calculated by

$$w_i = \begin{cases} e^{-ca_i} & \text{if } a_i < \pi/2 \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

where, i is view index, α_i is the angular distance of view I and w_i is weight for the view at that pixel. The constant c controls the fall off as the angular distance increases. Input views for which $\alpha_i \ge \pi/2$ are eli-

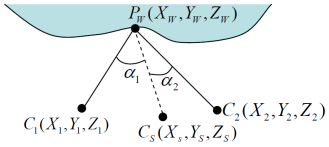


Figure 6. Weighted interpolation based on angular distances.

minated as they view the scene the other side. In practice, $c = 1 \text{ or } 2$ has been found to work well.

The new view is specified by

$$\begin{aligned}&\left[I_{\text{new}}, Z_{\text{new}}\right]\\&= \text{InitialView}\left(\left\{I_1, I_2, \cdots, I_N\right\}, \left\{Z_1, Z_2, \cdots, Z_N\right\}\right),\end{aligned} \tag{14}$$

where, *InitialView* is the procedure of pixel classification and initial new view creation as above described.

2.5. Find the Best Candidate for Unstable Pixel by Graph Cuts

In this step, we focus on refining initial synthetic view with unstable pixels. Unstable pixels have multiple pixel candidates and we want to predict the best candidate that minimizes the energy function described in following part.

We denote L as labeling space with $L = \{1, 2, \cdots, N\}$, representing the image index and let U be the set of unstable pixels. Let f_p be the label of unstable pixel p and $f_p \in L$. A labeling f is to assign a particular label f_p to a pixel $p \in U$. With this definition, our problem is to find the labeling f^* to fill the unstable region, such that the labeling f^* has minimum cost.

We define our energy function based on the Markov Random Fields (MRF) formulation:

$$E(f) = \sum_{p \in U} D_p\left(f_p\right) + \lambda \sum_{(p,q) \in N} V_{p,q}\left(f_p, f_q\right) \tag{15}$$

where, f is the labeling field, U is the set of unstable pixels, and N is the pixel's neighborhood system. $D_p\left(f_p\right)$ is called the data term, which defines the cost of assigning label f_p to pixel p. $V_{p,q}\left(f_p, f_q\right)$ denotes the smoothness term that evaluates the cost of disagreement between p and q which is assigned with f_p and f_q respectively. λ is a parameter to weigh the importance of these two terms.

Data term $D_p\left(f_p\right)$ is defined by

$$\begin{aligned}D_p\left(f_p\right) = \alpha Z_{f_p}(p) \sum_{q \in N_p} \left(1 - O_q\right)\left|I_{f_p}(p) - I_{\text{new}}(q)\right|\\ + \beta \sum_{i=1}^{N}\left|I_{f_p}(p) - I_i(p)\right|\end{aligned} \tag{16}$$

where N_p is neighboring pixels of p, $Z_{f_p}(p)$ is the depth value of pixel p at candidate f_p, $I_{\text{new}}(q)$ and O_q (0 or 1) are the color value and disoccluded indicator of pixel q, respectively. α and β are weight factors. $I_i(p)$ is color value of pixel p at input image i. $\left|I_i(p) - I_j(q)\right|$ represents the sum of absolute color differences between $I_i(p)$ and $I_j(q)$ of R, G and B channels.

The first part of data term enforces the candidate pixel selected to agree with its neighbor pixels. In addition, the neighboring pixel that is disocclusion does not influence the candidate selection process. It is also penalized less cost for the selecting a candidate pixel which has smaller depth value Z because the pixel with smallest depth value is closer to the camera and more likely defined the color of synthetic pixel p_2.

The second part of (16) is stationary cost, which defined based on color similarity at pixel p of all the input images. If the pixel p has similar color at more input images, the stationary cost is smaller.

Smoothness term $V_{p,q}\left(f_p, f_q\right)$: measures the penalty of two neighboring pixel p and q with different labels and is defined as follow:

$$V_{p,q}\left(f_p, f_q\right) = \frac{\left\|I_{f_p}(p) - I_{f_q}(p)\right\| + \left\|I_{f_p}(q) - I_{f_q}(q)\right\|}{2} \tag{17}$$

where, $\|*\|$ denotes the Euclidean distance in RGB color spaces. The smoothness term gives a higher cost if f_p and f_q do not match well. By incorporating such the smoothness term, we can achieve visually smooth in the synthetic image.

We apply graph cuts optimization that is public available in [9] to minimize our energy function $E(f)$. More detail about energy minimization with graph cuts can be found in [10,11].

This step is specified by

$$\begin{aligned}&I_{\text{new}}(U) = I_{f^*},\ Z_{\text{new}}(U) = Z_{f^*},\ \text{with } f^*\\&= \arg\min_f\left(E(f)\right),\end{aligned} \tag{18}$$

The refinement of image in **Figure 7** by using graph cut to select the best candidate for unstable pixel is shown in **Figure 8**.

2.6. Inpainting Disocclusion Pixels Based on the Depth and Color Values of Neighboring Pixels

Until this step, only the disocclusion regions are remaining. To deal with these disoccluded pixels, many papers such as [5,8] have developed algorithms based on the

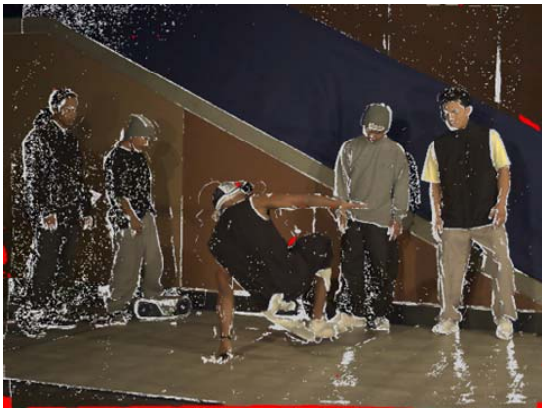


Figure 7. Initial synthesized view with 3 types of pixels. (The white color pixels are unstable pixels, the red color pixels are disoccluded pixels and the remaining pixels are stable pixels).

Figure 8. Refinement of initial synthesized image (image in Figure 7) by using graph cut (the red color pixels are disoccluded pixels).

inpainting method proposed by Tela [6]. Inpainting is a process of reconstructing lost or corrupted parts of images using the values of neighborhood pixels. Although, these algorithms work sufficiently well, the resulting inpainted regions contain a notable blur because of the mixture background and foreground colors at the edge of disoccluded regions. In this paper, we develop a technique based on inpainting method with depth information. We assume that the disoccluded pixels belong only to background, and we employ depth information to select accurately background pixels at the edges of disoccluded regions so that the blur can be avoided. Our method consists of several steps as follow.

First, for reducing processing time we find the small disoccluded regions by defining a window with the size of 3×3 centered at p and counting the unstable pixel inside this window. If the number of visible pixels M inside this window is higher than 50%, then the disoccluded pixels is inpainted by a weighted interpolation from visible pixels, which is specified by

$$I_{\text{new}}(p_{occ}) = \left(\sum_{i=1}^{M} d_i^{-1} * I_{new}(p_i)\right) \Big/ \sum_{i=1}^{M} d_i^{-1},$$
$$Z_{\text{new}}(p_{occ}) = \left(\sum_{i=1}^{M} d_i^{-1} * Z_{new}(p_i)\right) \Big/ \sum_{i=1}^{M} d_i^{-1}, \ \forall p_{\text{occ}} \in O, \quad (19)$$

where, M is number of visible pixels inside the window. O is disoccluded region, and d_i is distance from disoccluded pixel p_{occ} to visible pixel p_i. $I_{\text{new}}(p_i)$ and $Z_{\text{new}}(p_i)$ are color and depth values of the visible pixel p_i.

Second, for each pixel p_o in remaining disoccluded regions we search in eight directions to find the pixel p_u, which has the smallest depth value $Z_{\min}$ at the edge of disoccluded region and the distance d_u from this point to p_o. We define a window with the size of $(d_u+\Delta)\times(d_u+\Delta)$ centered at p_o (at first, $\Delta = 0$), and we count the visible pixels which have depth value Z with $|Z - Z_{\min}| \le 5$. If there are not enough 50% of visible pixels inside the window, we increase the size of window by increasing Δ. Finally, disoccluded pixels are inpainted by a weighted interpolation from visible pixels according to (19).

With inpainting procedure describing above, this step can summarized by

$$\left[I_{final}, Z_{final}\right] = \text{Inpaint}\left(I_{new}, Z_{new}\right). \quad (20)$$

3. Experimental Results

We quantify the proposal method performance based on Peak Signal Noise Ratio ($PSNR$) and the structural similarity (SSIM) index between a reference image I_r and a synthetic image I_s. SSIM index is a method for measuring the similarity between two images [12]. The SSIM index value 1 is only reachable when two images are identical and the higher PSNR normally indicates that it is higher quality synthetic image. Before computing $PSNR$, the images are converted from RGB color space to YUV color space, and Y channel is used for calculation. Y channel is defined by

$$Y(i,j) = 0.299R(i,j) + 0.587G(i,j) + 0.114B(i,j). \quad (21)$$

The $PSNR$ can be calculated by

$$PSNR = 10\log_{10}\left(\frac{255^2}{\frac{1}{w\cdot h}\sum_{i=0,j=0}^{w-1,h-1}\left\|Y_r(i,j) - Y_s(i,j)\right\|^2}\right), \quad (22)$$

where, w and h are the image width and height. Y_r and Y_s are the channels of reference image and synthetic image, respectively.

The proposed new view synthesis has been tested on "Break-dancer" and "Ballet" sequence which are gener-

ated and distribution by Interactive Visual Group at Microsoft Research [13]. These datasets include a sequence of 100 images of 1024×768 pixels captured from 8 cameras with the calibration parameters. **Figure 9** shows the camera arrangement of these two sequences. Depth maps for each view are also provided. For more detail about these depth maps generation, please refer to [2].

In our paper, the synthetic view is set to be the same as the actual camera. View 3 and 5 are used with depth maps to synthesize view 4. **Figure 10** shows the example of view synthesis results. The experimental results show that the proposed method achieved on average over 34 dB in PSNR and 0.93 index value in SSIM on the two sequence "Break-dancer "and "Ballet".

Figure 11 shows our PSNR and SSIM comparison with those of Sohl *et al.* [14] over 100 frames for the "Break-dancer" and "Ballet" sequences.

Because usually the number of cameras is limited, the camera arrangement is very importance for obtaining a good quality of synthesized view. **Figure 12** shows our quality of synthesis with varying the distance between the two reference cameras comparing the method presented by Mori *et al.* in [5], where our measurements

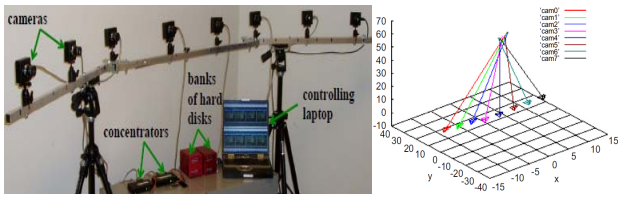
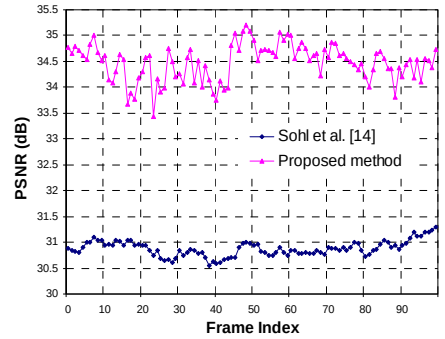


Figure 9. A configuration of "Break-dancer" and "Ballet" sequences with 8 cameras [2].

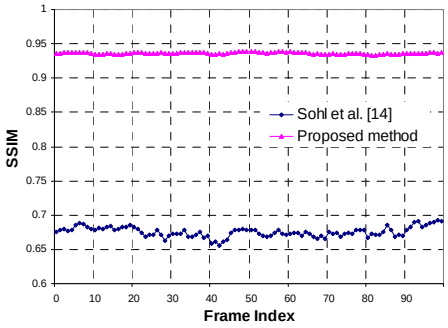
(a1) (b1)

(a2) (b2)

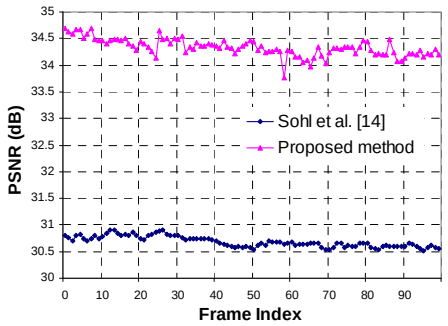
Figure 10. Example of the synthetic view. (a1) Original view image; (b1) Synthesized image (PSNR = 34.7 dB; SSIM = 0.94); (a2) Original view image; (b2) Synthesized image (PSNR = 34.6 dB; SSIM = 0.95).



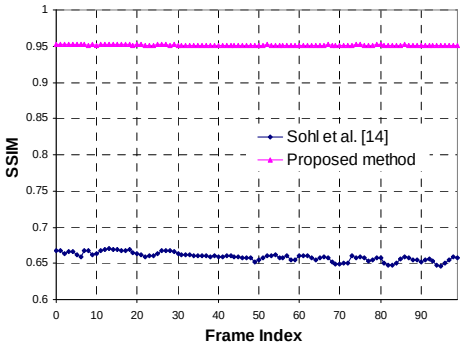
(a)



(b)



(c)

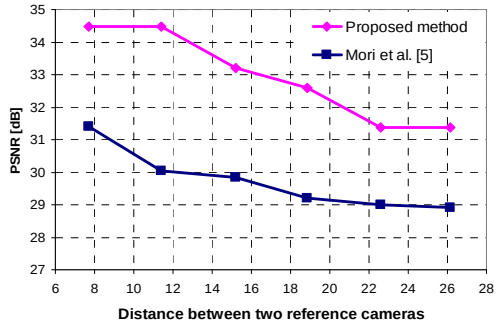


(d)

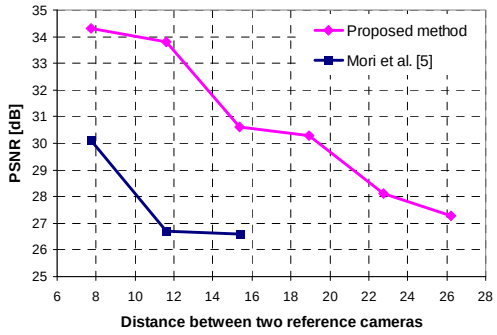
Figure 11. PSNR and SSIM comparison: (a) PSNR for "Break-dancer", (b) SSIM for "Break-dancer", (c) PSNR for "Ballet", (d) SSIM for "Ballet".

correspond to an average over 100 frames.

The measured synthetic qualities are compared with other methods and summarized in **Table 1**. From the



(a)



(b)

Figure 12. PSNR versus distance between camera for (a) "Break-dancer" sequence, (b) "Ballet" sequence.

Table 1. Exerimental results comparision.

Method	"Break-dancer"		"Ballet"	
	PSNR (dB)	SSIM	PSNR (dB)	SSIM
Sohl *et al*. [14]	30.8	0.68	30.7	0.66
Mori *et al*. [5]	31.4	Not Reported	30.1	Not Reported
Proposed method	**34.5**	**0.93**	**34.3**	**0.94**

results, the average PSNR of proposal is superior to that of other methods such as Mori *et al*. [5], Sohl *et al*. [14] with a gain of 3.0 dB. The structure similarity (SSIM) of our method is higher than that of Sohl *et al*. method.

Moreover, in multi-view configuration, we have N cameras, which capture the scene at difference positions. For our experimental case, there are 8 cameras. Thus, instead of using only two neighbor views as above conventional methods, we can use more than two images to synthesize a new view. Our proposal can do this idea easily. Our experiment shows that using four reference views (two views on both left side and right side) to synthesis a new view, a higher PSNR (about 0.5 - 1 dB) and SSIM are obtained than the case of using two reference views.

4. Conclusions

In this paper, we propose a novel synthesis method that enables to render a free-viewpoint from multiple existing cameras. The proposed method solves the main problems of depth based synthesis by performing the pixel classification to generate an initial new view from stable pixels and using Graph cuts to select the best candidate for unstable pixels. By defining the types of pixels and using Graph cuts, the color is consistent and the pixels are wrapped incorrectly because inaccuracy depth maps are removed. The remained disoccluded pixels are inpainted by using depth and texture neighboring pixel values. Considering depth information for inpainting, blurring between foreground and background textures are reduced. Experimental results show that the proposed method has strength in artifact reduction. In addition, our smooth term makes the result visually smooth. Objective evaluation has shown that our method gets a significant gain in PSNR and SSIM comparing to some other existing methods. Another advantage of our method is that we can use a set of un-rectified images in multi-view system to create a new view with higher quality.

The drawback of our method is using Graph Cuts, which is time consuming. However, we just only apply Graph Cuts for unstable pixels, which are a small amount of pixels comparing to the whole image, so the time for Graph Cuts can be reduced.

The future work will focus on more improving synthesis quality with utilizing temporal information in successive video frames.

5. Acknowledgements

We would like to thank Interactive Visual Media Group, Microsoft Research for distributing multi-camera video data and the anonymous reviewers for their comments.

REFERENCES

[1] M. Tanimoto, "Overview of FTV (Free-Viewpoint Television)," *Proceedings of the* 2009 *IEEE International Conference on Multimedia and Expo*, New York, 28 June-3 July 2009, pp. 1552-1553. doi:10.1109/ICME.2009.5202803

[2] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder and R. Szeliski, "High-Quality Video View Interpolation Using a Layered Representation," *ACM Transactions on Graphics*, Vol. 23, No. 3, 2004, pp. 600-608. doi:10.1145/1015706.1015766

[3] K. Pulli, M. Cohen, T. Duchamp, H. Hoppe, L. G. Shapiro and W. Stuetzle, "View-Base Rendering: Visualizing Real Objects from Scanned Range and Color Data," *Proceedings of the Eurographics Workshop on Rendering Techniques*'97, St. Etienne, 16-18 June 1997, pp. 23-24.

[4] K. Muller, K. Dix, P. Merkle, P. Kauff and T. Wiegand, "Intermediate View Interpolation Based on Multiview Video plus Depth for Advanced 3D Video Systems," 15*th IEEE International Conference on Image Processing*

(*ICIP*), San Diego, 12-15 October 2008, pp. 2448-2451

[5] Y. Mori, N. Fukushima, T. Yendo, T. Fujii and M. Tanimoto, "View Generation with 3D Warping Using Depth Information for FTV," *Signal Processing-Image Communication*, Vol. 24, No. 1-2, 2009, pp. 65-72. doi:10.1016/j.image.2008.10.013

[6] A. C. Telea, "An Image Inpainting Technique Based on the Fast Marching Method," *Journal of Graphics Tools*, Vol. 9, No. 1, 2004, pp. 25-36. doi:10.1080/10867651.2004.10487596

[7] S. Zinger, L. Do and P. H. N. de With, "Free-Viewpoint Depth Image Based Rendering," *Journal of Visual Communication and Image Representation*, Vol. 21, No. 5-6, 2010, pp. 533-541. doi:10.1016/j.jvcir.2010.01.004

[8] K.-J. Oh, S. Yea and Y.-S. Ho, "Hole Filling Method Using Depth Based In-Painting for View Synthesis in Free Viewpoint Television and 3-D Video," *Picture Coding Symposium*, Chicago, 6-8 May 2009, pp. 1-4.

[9] V. Kolmogorov and R. Zabih, "What Energy Functions Can Be Minimizedvia Graph Cuts?" *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 2, 2004, pp. 147-159. doi:10.1109/TPAMI.2004.1262177

[10] Y. Boykov, O. Veksler and R. Zabih, "Fast Approximate Energy Minimization via Graph Cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 23, No. 11, 2001, pp. 1222-1239. doi:10.1109/34.969114

[11] Y. Boykov and V. Kolmogorov, "An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 9, 2004, pp. 1124-1137. doi:10.1109/TPAMI.2004.60

[12] Z. Wang, A. C. Bovik and H. R. Sheikh, "Image Quality Assessment: From Error Measurement to Structural Similarity," *IEEE Transactions on Image Processing*, Vol. 13, No. 4, 2004, pp. 600-612. doi:10.1109/TIP.2003.819861

[13] S. M. Rhee, Y. J. Yoon, I. K. Shin, Y. G. Kim, Y. J. Choi and S. M. Choi, "Stereo Image Synthesis by View Morphing with Stereo Consistency," *Applied Mathematics & Information Sciences*, Vol. 6, 2012, pp. 195-200.

[14] M. Solh and G. AlRegib, "Hierarchical Hole-Filling for Depth-Based View Synthesis in FTV and 3D Video," *IEEE Journal of Selected Topics in Signal Processing*, Vol. 6, No. 5, 2012, pp. 495-504. doi:10.1109/JSTSP.2012.2204723

