Graph-based Interpolation for DIBR-synthesized Images with Nonlocal Means

Yu Mao *, Gene Cheung #, Yusheng Ji #
* The Graduate University for Advanced Studies, # National Institute of Informatics
Tokyo, Japan
{mao, cheung, kei}@nii.ac.jp

***Abstract*—Given texture and depth maps of a single reference viewpoint, one can synthesize a novel viewpoint image via depth-image-based rendering (DIBR) by mapping texture pixels from reference to virtual view. When the virtual viewpoint is much closer to the 3D scene than the reference view (camera movement in the z-dimension), objects close to the camera will enlarge in size in the virtual viewpoint image. An object's enlargement during DIBR means that its pixel samples in the reference view will be scattered to a larger spatial area, resulting in expansion holes. Following our previous work, we investigate the problem of expansion hole completion. We first assume a previously proposed method based on depth histogram is used to identify missing or erroneously translated pixels as expansion holes. We then propose a new graph-based interpolation technique to fill in expansion holes. Unlike our previous work, nonlocal but similar pixel patch information are incorporated into a new graph construction before a graph-based interpolation procedure with sparsity prior is executed, resulting in enhanced performance. Experimental results show that our new procedure of expansion hole filling can outperform inpainting procedure employed in VSRS 3.5 by up to 4.02dB.**

I. Introduction

Recent advances in depth sensing technologies such as Microsoft Kinect means that depth maps (per-pixel distance between captured objects in the 3D scene and capturing camera) can now be affordably captured along with color images (texture maps). Given texture and depth maps from the same camera view, a user can synthesize a new virtual viewpoint image via *depth-image-based rendering* (DIBR) [1]: each texture pixel in the reference view is mapped to a pixel location in the virtual view, where the mapped location is derived from the corresponding depth pixel in the reference view. Missing pixels in the virtual view (*disoccluded* spatial locations that were not visible in the reference view) are completed using depth-based inpainting algorithms [2]. For small camera motion along the x- or y-dimension, this approach of DIBR synthesis plus inpainting works reasonably well.

If the camera motion implied by the chosen virtual viewpoint is along the z-dimension, however, the view rendering process becomes more complex. z-dimensional camera motion is possible, for example, in immersive applications such as video conferencing, where a viewer observes real-time synthesized images on a 2D display, whose rendering perspective changes according to the tracked head position of the viewer. An observer's head moving back and forth then corresponds to z-dimensional camera movements. When the virtual viewpoint is located much closer to the 3D scene than the reference view, object near the camera will enlarge in size in the virtual view. An enlargement in object size means that pixel samples of the object surface in the reference view will be scattered to a larger spatial area during DIBR, resulting in *expansion holes*. We focus on the problem of expansion hole completion in the DIBR-synthesized image in this paper.

Specifically, we propose the following two-step procedure for expansion hole filling in DIBR-synthesized images. We first assume our previously proposed method based on depth histograms [3] is executed to identify missing or erroneously synthesized pixels as expansion holes. We then propose a new graph-based interpolation technique to fill in expansion holes. Unlike our previous work [3] that leverages only on local patch information for interpolation, we incorporate in addition nonlocal but similar patch information (similar to *nonlocal means* algorithm (NLM) in [4] for image denoising) into a new graph construction, before a graph-based interpolation procedure with sparsity prior is executed, resulting in enhanced performance. Experimental results show that our new procedure of expansion hole filling can outperform inpainting procedure employed in VSRS 3.5 by up to 4.02dB.

The structure of the paper is as follows. We first discuss related work in Section II. We then overview our DIBR view synthesis system in Section III. We overview our previously proposed methodology to identify expansion holes in the virtual view in Section IV, and discuss our graph-based interpolation technique in Section V. Finally, experimentation and conclusions are presented in Section VI and VII, respectively.

II. Related Work

Texture-plus-depth format [5]—representation of a 3D scene in texture and depth maps from one or more viewpoints—can enable low-complexity rendering of freely chosen viewpoint images at decoder via DIBR [1]. We assume a virtual viewpoint image is synthesized from just one texture / depth map pair from one single camera view, which has been shown in previous work [6] to lead to good rate-distortion (RD) performance, assuming that the larger disocclusion holes in the synthesized image can be inpainted appropriately. Instead of disocclusion holes, the focus of this paper is on expansion hole filling due to large z-dimensional camera motion.

Expansion hole filling can be posed as a super-resolution (SR) problem, solved using conventional image SR algorithms [7] on rectangular pixel grid. For example, texture and

depth maps in the reference view can be super-resolved into a finer rectangular grid of sufficiently high resolution (one where all possible expansion holes of original resolution in the virtual view will be covered), then performing DIBR to see which of the super-resolved pixels land on the virtual view pixel grid. Unlike this SR approach which requires computation of a possibly very large number of pixels in the reference view (and only a smaller subset get mapped to the grid points in the virtual view), our approach is a *parsimonious* one: only grid samples identified as expansion hole pixels in the virtual view are interpolated, leading to lower complexity relative to the aforementioned SR approach.

We advocate an image interpolation method based on *graph-based transform* (GBT), which uses the eigenvectors of a defined graph Laplacian matrix to provide a Fourier-like frequency interpretation. Unlike previous fixed transform based interpolation like DCT [8] defined on rectangular pixel grid, GBT is adaptive to a more general setting where any n unknown pixels can be interpolated using any m known pixels, all connected via a weighted graph. Further, unlike our previous work [3] where only local information are used for interpolation, we leverage on the self-similarity characteristic of natural images (as done in NLM [4]) and search for nonlocal similar patches when constructing a graph for better performance. We will show in Section VI that leveraging nonlocal information does improve performance over [3].

III. Interactive Free-viewpoint System

We first describe a system model for our interactive free viewpoint streaming system. The server transmits a texture / depth map pair from one *reference* camera viewpoint to the client, so that the client can freely select a virtual viewpoint near the camera viewpoint for DIBR-based image rendering of the 3D scene. Selecting a virtual viewpoint far away from the reference viewpoint will trigger server's transmission of a new texture / depth map pair of a different reference view. In this paper, we focus only on synthesis of virtual view images in the neighborhood with large z-dimensional camera movements.

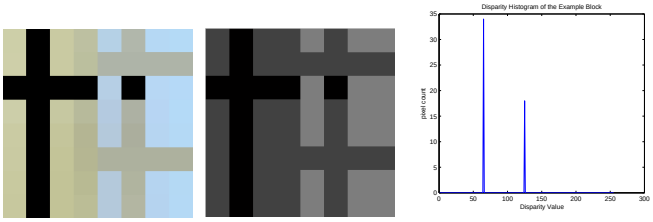
A. Hole Filling in DIBR Synthesized Image

After 3D warping, we likely observe *holes* in the virtual view, i.e., a pixel in the virtual view that has no corresponding pixel in the reference view. There are two main kinds of holes. The first kind is *disocclusion holes*: the corresponding spatial region in the reference view is occluded by an object closer to the camera, but become exposed after projecting to the virtual view. Disocclusion holes can be filled via depth-based image inpainting techniques [2], and are outside the scope of this paper. The second kind is *expansion holes*. We define an expansion hole as follows: a spatial area of an object's surface in the virtual view, whose corresponding area in the reference view is visible but smaller in size. Unlike disocclusion holes, expansion holes can leverage on information of neighboring pixels with similar depth (indicating they are of the same object) for interpolation.

We first identify pixels in the virtual view as expansion holes using a method based on depth histogram, as done in [3]. We then propose a graph-based interpolation procedure with a sparsity prior for expansion hole filling.

IV. Expansion Hole Identification

We briefly review the procedure in [3] to identify expansion holes in the DIBR-synthesized virtual view image. Denote (x, y) the coordinate of a pixel in the reference view. When rendering from reference view to virtual view, a projection function $\mathcal{F}(x, y) = (x', y')$ maps a pixel (x, y) in reference view to location (x', y') in virtual view. The inverse projection function $\mathcal{F}'(x', y') = (x, y)$ maps from (x', y') in virtual view to (x, y) in reference view. Both $\mathcal{F}$ and $\mathcal{F}'$ can be easily derived from standard 3D warping equations [1].



(a) texture block (b) disparity block (c) depth histogram

Fig. 1. Example of texture / disparity block and constructed depth histogram.

We first divide the virtual view into blocks of $b \times b$ pixels. For a given block, we decompose it into *depth layers* as follows: i) construct a histogram of depth values of the synthesized pixels in the block, ii) separate depth pixels into layers by identifying local minima in the histogram and using them as layer-dividing boundaries. Fig. 1 shows an example texture and disparity[1] block, and corresponding depth histogram. We next process each layer in order of increasing depth values (closest layer to the camera first). When processing a layer l, all synthesized pixels of high layers $l + 1, \ldots$ are treated as empty pixels; this allows us to erase a synthesized background pixel during expansion hole filling of a foreground object.

We examine each empty pixel in the block as follows. As shown in Fig. 2, we divide the neighborhood of an empty pixel (marked X in Fig. 2) into four quadrants. In each quadrant, we find an available pixel that is closest to our target empty pixel

[1]There is a one-to-one correspondence between depth and disparity, where disparity is inversely proportional to depth. Thus, disparity map can be equivalently processed instead of depth map.

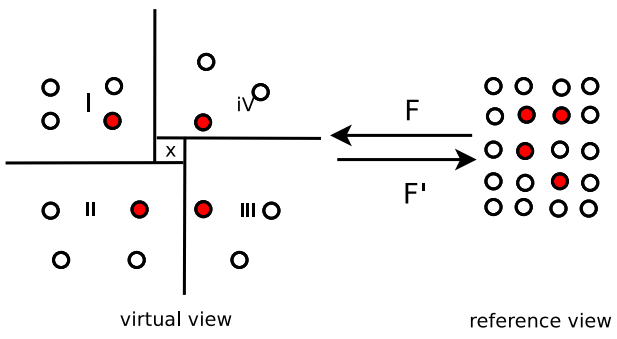


Fig. 2. Expansion Holes on a Depth Layer

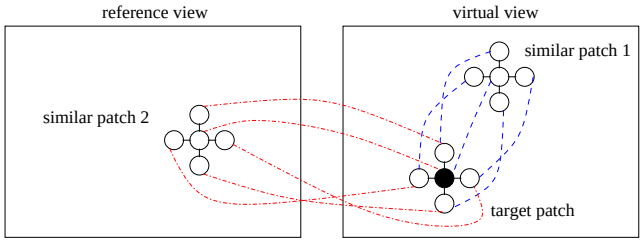


Fig. 3. Connecting pixels of similar patches to target patch

(if one exists). We then map these closest pixels back to the reference view using $\mathcal{F}'$, to see if any pairs are *neighboring* pixels, i.e., their Euclidean distance is less than a threshold n. If two or more pairs of neighboring pixels are found, we declare the target empty pixel is an expansion hole pixel.

The intuition behind this method is that if the set of closest pixels in the virtual view are neighboring pixels in the reference view, then the empty pixel in the virtual view is very likely to be inside the convex set spanned by the closest pixels in the reference view. After identifying empty expansion hole pixels, we interpolate them using a graph-based interpolation method described next.

V. Expansion Hole Pixel Interpolation

We now discuss how we fill expansion hole pixels using a graph-based interpolation procedure with sparsity prior. We first discuss how GBT basis functions are derived from a carefully constructed graph. We then discuss how the optimization is formulated and performed given the derived GBT basis functions.

A. Constructing a Graph-based Transform

We first discuss how to construct a graph $\mathcal{G}$ connecting correlated pixels locally and globally. First, among expansion hole pixels of the first unprocessed depth layer in an image, we identify a $b\times b$ *target patch* P_t with a missing pixel at its center (*centroid*) c_t, where P_t has the fewest number of missing pixels among patches of same size. We then globally search for *similar patches* P_s's of the same size as P_t, where by similar we mean the l_2-norm of the pixel-wise patch difference $\|P_t - P_s\|_2$ is no larger than a threshold τ. As shown in Fig. 3, a similar patch can be nonlocal from the same virtual view image (as done in NLM [4]), or from the reference view image. To control the complexity required, the search is done via random sampling [9] with a Gaussian kernel centered at the target patch P_t.

Given a small set of similar patches P_s's, we draw an edge between corresponding pixels i in P_t and j in P_s, where the edge weight $e_{i,j}$ is a multiplication of three terms:

$$e_{i,j} = w_{i,j}\, u_{i,j}\, v_{i,j} \tag{1}$$

$w_{i,j}$ is the difference in *relative spatial distance* between i in P_t and j in P_s, i.e., difference in distance from each pixel to its respective patch centroid. For example, if pixel i in target P_t is d_i from centroid c_t and its corresponding pixel j in P_s is d_j from centroid c_s, then $w_{i,j}$ is:

$$w_{i,j} = e^{-\frac{\|d_i - d_j\|_2}{\sigma_d^2}} \tag{2}$$

where σ_d^2 is a parameter to control the sensitivity of $w_{i,j}$ to the distance difference.

$u_{i,j}$ is the *photometric* difference between pixel i in P_t and pixel j in P_s, i.e., the pixel intensity difference in exponential form as written in (2) with parameter σ_r^2. An empty expansion hole pixel needs an intensity value for $u_{i,j}$ to be properly defined; we simply copy the corresponding pixel value from the most similar patch over for the sake of defining $u_{i,j}$, as done in [10]. $w_{i,j}$ and $u_{i,j}$ constitute the two considerations (spatial and photometric distances) typically used in local image filtering such as bilateral filter [11]. $v_{i,j}$ measures the patch-level similarity between P_t and P_s, similarly done in other nonlocal methods [4]. Fig. 3 shows an example where a target patch of 5 pixels is connected to two similar patches.

We also draw local edges between centroid and pixels in the same patch, where the edge weight is composed only of two terms: $w_{i,j}$ and $u_{i,j}$. $w_{i,j}$ is then the difference in *absolute spatial difference*, as done in [11].

Having constructed a graph $\mathcal{G}$ for target and similar patches, we next overview the procedure to construct a GBT [12], which is a signal-adaptive block transform. We first define the degree matrix $\mathbf{D}$ and adjacency matrix $\mathbf{A}$ from the graph $\mathcal{G}$. Adjacency matrix $\mathbf{A}$ has entry $A_{i,j}$ containing edge weight $e_{i,j}$ if edge connecting i and j exists, and 0 otherwise. Degree matrix $\mathbf{D}$ is a diagonal matrix with non-zero entries $D_{i,i} = \sum_j e_{i,j}$. A *graph Laplacian* $\mathbf{L} = \mathbf{D} - \mathbf{A}$ can then be defined. Finally, we perform eigen-decomposition on $\mathbf{L}$, i.e., find eigenvectors ϕ_i's such that $\mathbf{L}\phi_i = \rho_i\phi_i$, where ρ_i is the i-th graph frequency. The basis vectors of the GBT are ϕ_i's.

B. Linear Program Formulation

Without loss of generality, let the N synthesized (known) pixels in $\mathcal{G}$ be $s_1, \ldots, s_N$, and the interpolated length-M signal, $M > N$, be $\hat{\mathbf{s}} = [\hat{s}_1, \ldots, \hat{s}_M] = \Phi\mathbf{w}$, where columns of $M\times M$ matrix Φ, ϕ_j's, are the M GBT basis vectors as derived earlier, and $\mathbf{w}$ is the code vector for signal interpolation. Let $\mathbf{u}_i$'s be a set of N length-M unit vectors, $[0, \ldots, 0, 1, 0, \ldots, 0]$, where the single non-zero entry is at position i. Our objective is to minimize a weighted sum of: i) the l_1-norm[2] of the difference between interpolated signal $\hat{\mathbf{s}}$ and original signal $\mathbf{s}$ at the N synthesized pixel locations, and ii) weighted l_0-norm of the code vector $\mathbf{w}$:

$$\min_{\mathbf{w}} \sum_{i=1}^{N} \|\mathbf{u}_i^T\Phi\mathbf{w} - s_i\|_1 + \lambda\|\mathbf{w}\|_0 \tag{3}$$

As typically done in the literature for sparse signal recovery, we replace the l_0-norm above with a l_1-norm for ease of computation:

$$\min_{\mathbf{w}} \sum_{i=1}^{N} \|\mathbf{u}_i^T\Phi\mathbf{w} - s_i\|_1 + \lambda\|\mathbf{w}\|_1 \tag{4}$$

[2] l_1-norm is chosen here for complexity reason; the derived minimization can then be computed efficiently using linear programming.

TABLE I
PSNR COMPARISON FOR EXPANSION HOLE FILLING.

method	VSRS+	GBT	NLGBT
art PSNR(dB)	19.56	23.36	23.58
moebius PSNR(dB)	19.47	23.15	23.33

(4) can now be easily rewritten as a linear programming (LP) formulation, and can thus be solved using any known LP algorithms. Then the interpolated signal will be used to update the graph weights. This procedure will be carried out iteratively until convergence.

VI. EXPERIMENTATION

A. Experimental Setup

We used art and moebius in Middlebury's 2005 datasets[3] as our multiview image test sequences. We use the same methodology in [3] to first generate a reference view v_r with texture and depth maps of lower resolution than captured images. Using texture and depth maps of v_r, we used DIBR to generate virtual view v_0. We used one of the following three methods for filling of expansion holes. In the first method we call VSRS+, we modified VSRS software version 3.5 to use a single reference view. We note that the VSRS software is not designed for synthesis of virtual view images with significant z-dimensional camera movements. NLGBT is our proposed scheme in this paper, and GBT is the algorithm in [3].

B. Experimental Results

We calculated the PSNR of the virtual view images interpolated using the aforementioned three interpolation methods against the ground truth v_0. We only calculated the PSNR of the identified expansion hole areas. The PSNR comparison is shown in Table I. For the art sequence, we see that both GBT and NLGBT outperformed VSRS+ significantly: by 3.8dB and 4.02dB, respectively. This demonstrates that the correct identification of expansion holes and subsequent interpolation are important for DIBR image synthesis of virtual view with significant z-dimensional camera movement. Further, we see that NLGBT outperformed GBT by 0.18dB, showing that using nonlocal information, we can achieve better image quality. For the moebius sequence, we observe a similar trend.

(a) holes (b) VSRS+ (c) NLGBT

Fig. 4. Visual comparison between VSRS+ and NLGBT for sequence art.

Next, we examine the generated image quality visually. In Fig. 4, we show an example patch of the DIBR-synthesized image before filling of expansion holes, and after filling of expansion holes using VSRS+ and NLGBT, respectively, for the art sequence. The resolution of the patch is 200×200. First, we see visually in Fig. 4(a) that the presence of expansion holes is everywhere and is a significant problem. Note also that the nature of expansion holes is very different from disocclusion holes (e.g., right of the brush), which are larger contiguous regions. Second, we see in Fig. 4(b) that applying inpainting algorithm naïvely to fill in all missing pixels indiscriminately does not lead to acceptable quality for expansion hole areas. Finally in Fig. 4(c) the image shows that NLGBT can achieve better performance in texture interpolation with the help of nonlocal information.

[3]http://vision.middlebury.edu/stereo/data/scenes2006/

VII. CONCLUSION

When the observer's chosen virtual viewpoint for image rendering via DIBR involves large camera motion in the z-dimension relative to the reference viewpoint (camera moving closer to the 3D scene), objects closer to the camera will increase in size significantly. Because insufficient number of pixel samples are available in the reference image, expansion holes in the DIBR-rendered image will appear. In this paper, we propose a graph-based interpolation procedure to fill in expansion holes. Unlike our previous work, we search for nonlocal but similar pixel patches and incorporate them into a new graph construction, before performing a graph-based interpolation procedure with sparsity prior. Experimental results show that our new procedure of expansion hole filling can outperform inpainting procedure employed in VSRS 3.5 by up to 4.02dB.

REFERENCES

[1] D. Tian, P.-L. Lai, P. Lopez, and C. Gomila, "View synthesis techniques for 3D video," in *Applications of Digital Image Processing XXXII, Proceedings of the SPIE*, vol. 7443, 2009, pp. 74 430T–74 430T–11.

[2] K.-J. Oh, S. Yea, and Y.-S. Ho, "Hole-filling method using depth based in-painting fore view synthesis in free viewpoint television (FTV) and 3D video," in *Picture Coding Symposium*, Chicago, IL, May 2009.

[3] Y. Mao, G. Cheung, A. Ortega, and Y. Ji, "Expansion hole filling in depth-image-based rendering using graph-based interpolation," in *IEEE International Conference on Acousitics, Speech and Signal Processing*, Vancouver, Canada, May 2013.

[4] A. Buades, B. Coll, and J. Morel, "A non-local algorithm for image denoising," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 2005)*, San Diego, CA, June 2005.

[5] P. Merkle, A. Smolic, K. Mueller, and T. Wiegand, "Multi-view video plus depth representation and coding," in *IEEE International Conference on Image Processing*, San Antonio, TX, October 2007.

[6] I. Daribo, G. Cheung, T. Maugey, and P. Frossard, "RD optimized auxiliary information for inpainting-based view synthesis," in *3DTV-Conference*, Zurich, Switzerland, October 2012.

[7] X. Li and M. Ochard, "New edge-directed image interpolation," in *IEEE Trans. Image Processing*, vol. 10, no.10, October 2001, pp. 1521–1527.

[8] Z. Alkachouh and M. Bellanger, "Fast DCT-based spatial domain interpolation of blocks in images," in *IEEE Transactions on Image Processing*, vol. 9, no.4, April 2000.

[9] S. H. Chan, T. Zickler, and Y. M. Lu, "Fast non-local filtering by random sampling: it works, expecially for large images," in *IEEE International Conference on Acousitics, Speech and Signal Processing*, Vancouver, Canada, May 2013.

[10] C. Hu, L. Cheng, and Y. M. Lu, "Graph-based regularization for color image demosaicking," in *IEEE International Conference on Image Processing*, Orlando, FL, September 2012.

[11] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proceedings of the IEEE International Conference on Computer Vision*, Bombay, India, 1998.

[12] G. Shen, W.-S. Kim, S. Narang, A. Ortega, J. Lee, and H. Wey, "Edge-adaptive transforms for efficient depth map coding," in *IEEE Picture Coding Symposium*, Nagoya, Japan, December 2010.