SELECTIVE HOLE-FILLING FOR DEPTH-IMAGE BASED RENDERING

Adriano Oliveira, Guilherme Fickel, Marcelo Walter, Cláudio Jung

Institute of Informatics, Federal University of Rio Grande do Sul

ABSTRACT

One of the biggest challenges in view interpolation is to fill the regions without projective information in the synthesized view. In this paper, we present a new approach that identifies and corrects different types of missing information. In the first stage, we propose a fast solution to tackle the problems of cracks and ghost, common artifacts in the view interpolation process. Then, we complete larger holes by exploring the disparity map as an additional cue to select the best patch in a patch-based inpainting procedure. Our experimental results indicate that we were able to outperform current state of the art hole filling techniques for view interpolation.

Index Terms— View interpolation, hole filling, DIBR, view synthesis, inpainting

1. INTRODUCTION

With the recent development of 3D displays, much more content is now generated using multiple cameras. This fostered the research of DIBR (Depth Image Based Rendering) techniques, which consists of using a single reference image (usually from the left camera) with its respective disparity map to generate another synthesized view, usually a reconstruction of the second camera.

DIBR techniques are of great importance for stereo video encoding since they can greatly reduce the required bandwidth by only encoding one color (reference) image and its respective grayscale disparity map, instead of two color images. And by having the ability to generate other synthesized views with arbitrary baselines, many applications may be possible such as free viewpoint camera where the spectator can change in real time the desired view of the scene, baseline retargeting to adapt the stereo baseline depending on the characteristics of each display, etc.

However, the generation of interpolated views presents several challenges, and visual artifacts are common. In this paper we directly deal with three of the most common classes of visual artifacts: cracks, which are generated due to the quantization of the disparity map, ghosts, which occur when we have a disparity discontinuity that is not well defined in the image domain, and holes, that are larger areas of unprojected data due to occlusions and/or errors in the disparity map. The pipeline of our approach can be seen in Figure 1.

2. RELATED WORK

The crack holes are long and thin, usually 1 to 2 pixels wide. To solve this problem some techniques estimate the disparity of the synthetic view and fill the cracks in the disparity with some simple filtering procedures [1, 2], such as a median filter. Another solution is to apply a simple filtering or inpainting algorithm directly to the synthetic view cracks [3].

The holes caused by disocclusions, on the other hand, are usually large both in length and width. Inpainting methods that propagate information through diffusion [4], even if they propagate incoming edges such as [5], are not able to propagate complex textures. To coherently fill those holes a more robust solution is needed. Mori and colleagues [2] proposed to project both the left and right views in the synthetic view position. Both views are combined using an alpha blending procedure, and the holes from one projection are mostly completed by the information of the other. The remaining holes are usually small, and a simple inpainting algorithm can be used to estimate them. However, since this procedure uses both stereo views and depth maps, it is not suitable for DIBR.

An interesting inpainting solution with texture synthesis was proposed by Criminisi and colleagues [6], which fills the holes by copying patches from the available image. They showed that by filling the holes within a certain order, prioritizing first complete regions that had strong edges intersected with the hole, their approach was able to correctly synthesize textures while propagating the edges. By recognizing that holes generated by occlusion typically belong to the background, Daribo and Saito [7] proposed to change the priority function from [6] to fill the holes starting from the background. By using the disparity information, they were able to correctly propagate the background texture information to fill the holes and achieve more coherent results. Oh and colleagues [1] also proposed to use the disparity information in the inpaint procedure. They adapted the inpaint order from Telea's algorithm [8], but the results are not very good in large areas. This is due to the limitations of Telea's algorithm that, similarly to Bertalmio's approach, propagates color information as well as edges but is not able to propagate complex textures. Hervieu and colleagues [9] presented a two-stage process: in the first one, the disparity map is inpainted, and used as a basis to inpaint the stereo pair using an extension of [6]. Mao et al. [10] presented an approach for

978-1-4673-6997-8/15/$31.00 ©2015 IEEE

identifying expansion holes, and two methods for correcting them: the first one, based on linear interpolation, is very simple and fast; the second one, based on graphs with a sparsity prior, is better but more expensive computationally.

A different approach was developed by Solh and AlRegib [11], called Hierarchical Hole-Filling (HHF). They produce pyramid-like lower resolution estimates of the synthetic view with holes by taking the mean between blocks of 5×5 of the valid pixels (i.e. ignoring pixels without projection information), and propagating it to one pixel in the next scale. Within a few multi-resolution scales they obtain a low resolution estimative of the synthetic view without holes. By propagating this low resolution image along the multi-scale structure they estimate the holes in the original image.

Solh and AlRegib also proposed to use a pre-processed image as the input for the HHF algorithm. This algorithm is called depth adaptive HHF, and the pre-processed image is the synthetic view with holes weighted according to the disparity. The main idea is to give a higher importance (weight) for lower disparity regions since they belong to the background, and holes ideally should be completed by background pixels. However the depth adaptive HHF has a minor impact in the final results over the original HHF.

3. THE PROPOSED APPROACH

3.1. Cracks Removal

To tackle the artifacts caused by cracks, our first step is to correctly identify them. We compute a binary image S that contains all the pixels in the synthetic view that do not have any projection information. Then, S is filtered with a morphological opening operation using a structuring element H_C, resulting in a filtered image called $\hat{S}$. In this work, we used H_C with a horizontal line format, with length of 1 pixel and width of 2 pixels, and the inverse is used to verify vertical small holes.

Image $\hat{S}$ fills out thin vertical lines of S, so that the binary mask C containing all the cracks can be found by:

$$C = S \setminus \hat{S}, \tag{1}$$

with $\setminus$ being the absolute complement operator in set terminology. The result of this identification process can be seen in the block diagram (Fig. 1), where the cracks are painted red in the images that illustrates the "fill cracks" stage.

After identifying all the cracks in the synthesized image, we use a fast inpainting procedure proposed by Oliveira and colleagues [4]. Let Ω be the crack to be inpainted and $\partial\Omega$ its boundary, the inpainting procedure is approximated by an isotropic diffusion that propagates the information from $\partial\Omega$ to Ω. Initially, the color information of Ω is cleared and the diffusion process is approximated by repeatedly convolving the region to be inpainted with the diffusion kernel shown in Fig. 2.

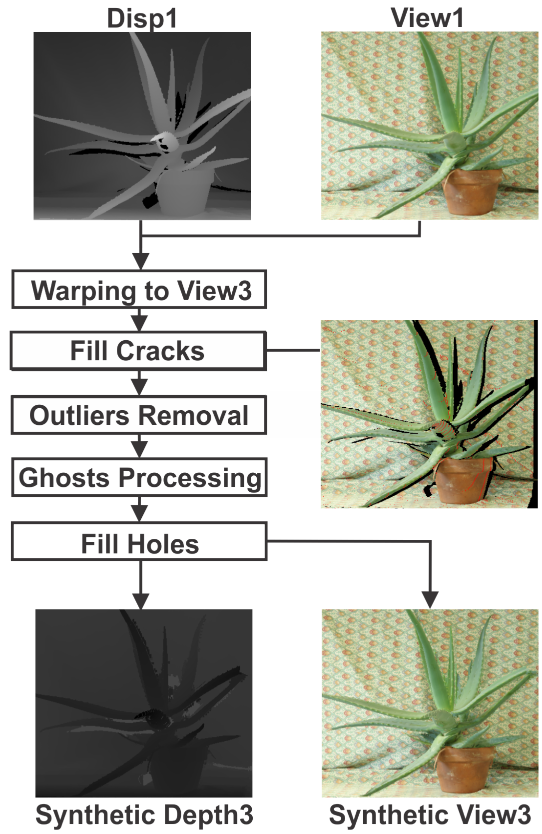


Fig. 1. Block diagram of the proposed algorithm.

a	b	a
b	0	b
a	b	a

Fig. 2. Diffusion kernel used, with $a = 0.073235$ and $b = 0.176765$.

This simple approach may introduce blurring when Ω crosses the boundaries of high contrast edges. In practice, however, the cracks are usually only 1 or 2 pixels wide (as defined by the structuring element), so only a small number of iterations is needed, and the resulting blurring artifacts are not noticeable.

Additionally, we also detect small isolated "islands" of projected pixels within holes of unknown data, typically due to errors in the disparity map. These outliers are identified and removed using morphological opening with linear structuring elements (1×2 for horizontal outliers, and 3×1 for vertical ones).

3.2. Ghosts Removal

Due to the finite size of image sensors and imprecisions of the disparity map, pixels around an image boundary are usually composed by the foreground and background objects. The ghost artifacts consist in foreground information being propagated to background regions due to the lack of information of the disparity map in representing those smooth boundaries. It is important to notice that those artifacts can greatly impact the inpaint algorithm, so it is necessary to deal with them first.

In order to identify the regions potentially related to ghosts, we calculate the binary image G that is computed by excluding the crack regions C from S as $G = S - C$. After we have the regions G that may contain ghosts, we use the morphological dilation operator with a non-symmetric structural element H_G to expand the occluded regions in the direction of the reference camera to the virtual camera. For example, if we have one reference camera and generate a synthetic view using a virtual camera on the right side, the background information of all the holes caused by occlusion problems will be on their right side. Therefore, depending on the side, a different mask H_G is generated. In all of our tests, the structural element H_G used was a horizontal line, one pixel tall and 3 pixels wide, whose configuration varies depending on the projection orientation (left or right). Then we separate candidates using the absolute complement operator in the processed image $\hat{G}$, obtaining σ_Ω, as in the cracks removal approach. Fig. 3 illustrates this process.

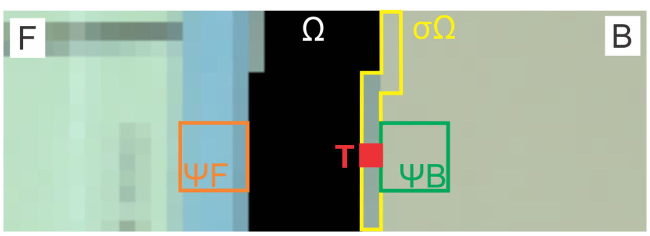


Fig. 3. Zoom of monopoly dataset [12]. Notation: σ_Ω are ghost candidates. Ω, F and B represent the hole, foreground and background respectively. ψ_B and ψ_F are patches for evaluation of the target (T) similarity with F and B.

The next step is to evaluate each candidate point $T \in \sigma_\Omega$ based on its similarity with neighboring patches. For that purpose, we compute the mean intensity within 3×3 patches ψ_B and ψ_F, which are neighbors of T in the background and foreground, respectively, and compute the differences dB and dF from T to these mean values. If $dB < dT$ and $dB < \alpha$, where T is a similarity threshold, then T is kept attached to the background. Otherwise, it is considered as belonging to the foreground, and moved horizontally to the other extremity of the hole. In our tests, we used $\alpha = 11$ as the threshold in all experiments.

3.3. Hole Filling

Due to the sampling theorem, there are constraints to the spatial frequency content of an image that cannot be reconstructed once lost. In those cases of missing or damaged areas, the best we can achieve is to produce a plausible result rather than a perfect reconstruction [4].

Since it is not uncommon to have big holes in DIBR techniques caused by occlusions and/or disparity problems, we need to synthesize not only a plausible color within the holes but also to recreate a locally adequate texture. For this task we propose to extend the texture synthesis work of [6] to prioritize the background regions in the hole filling algorithm, since the occluded regions are by definition a portion of the background that has been disoccluded.

Given a hole Ω and its boundary $\partial\Omega$, the first step is to find the patch Ψ_p with $p \in \partial\Omega$ that must be inpainted. We then search for a patch Ψ_q in the source region $\Phi = \mathcal{I} - \Omega$, where $\mathcal{I}$ is the image to be inpainted, and copy its texture to Ψ_p. The main idea is to use Φ as a texture database, and copy small patches Ψ_q to Ω according to the local information provided by Ψ_p.

The first step is to define the hole filling order, aiming to both preserve incoming edges and prioritize the background. The choice of p for each iteration is given by the following priority equation $P(p) = C(p)E(p)$ where $P(p)$ is the priority for a given pixel $p \in \partial\Omega$, $C(p)$ is the confidence term (described in [6]) and $E(p)$ is the depth term. They are defined as follows:

$$C(p) = \frac{\sum_{q \in \Psi_{p \cap (\mathcal{I} - \Omega)}} C(q)}{|\Psi_p|}, \tag{2}$$

$$E(p) = \frac{\sum_{q \in \Psi_{p \cap (\mathcal{I} - \Omega)}} d(q)}{|d(p)|}, \tag{3}$$

where $|\Psi_p|$ is the area of Ψ_p, $|d(p)|$ is the area of $d(p)$ (in terms of number of pixels), $d(q)$ is the depth value for each point of the patch. The priority $P(p)$ is calculated for every $p \in \partial\Omega$ for each iteration, and the point with the biggest value is chosen. In the initial configuration, $C(p)$ is set to zero $\forall p \in \Omega$, and $C(p) = 1\ \forall p \in \mathcal{I} - \Omega$.

The confidence term $C(p)$ measures the amount of reliable information around the pixel p, so it prioritizes filling patches which have more pixels already filled. The depth term $E(p)$ prioritizes the greatest depths, which naturally favor background pixels.

After choosing the destination patch Ψ_p to be filled, the last step is to find the origin patch Ψ_q that is obtained from Φ. We choose Ψ_q by searching in Φ the patch that is the most similar to Ψ_p:

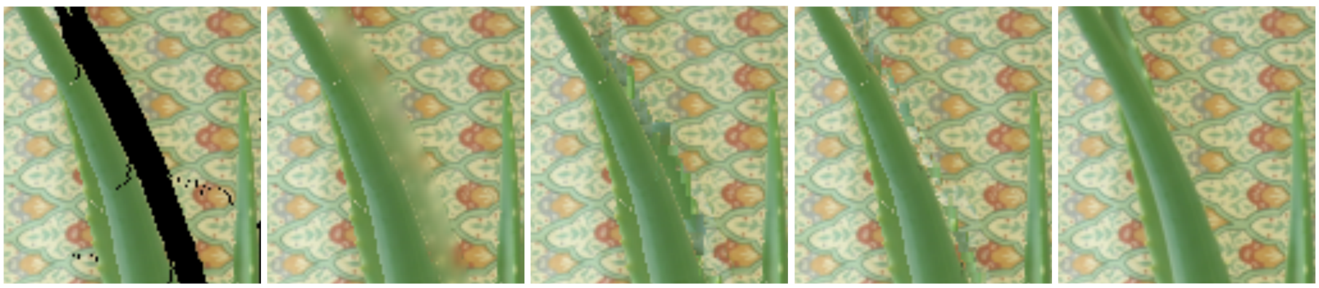
$$\Psi_q = \arg \min_{\Psi_q \in \Phi} s(\Psi_p, \Psi_q), \tag{4}$$

$$s(\Psi_p, \Psi_q) = \sum_{x \in \Omega_v(\Psi_p)} \|\Psi_p(x) - \Psi_q(x)\|^2,$$

where $\Omega_v(\Psi_p)$ denotes the set of pixels in Ψ_p containing valid information, $\Psi_p(x)$ is the RGB color vector related to pixel x.

Method	Aloe1	Aloe5	Art1	Art5	Books1	Books5	Monopoly1	Monopoly5	Mean
Criminisi	26.8196	26.9094	23.6227	23.7331	27.2535	29.3389	27.8316	23.6397	26.1436
HHF	26.7166	27.5551	24.1260	24.9558	27.7551	29.2626	29.2352	26.7045	27.0389
Proposed	**27.4329**	**27.7281**	**26.3628**	**25.2687**	**29.6710**	**29.7081**	**29.6776**	**28.7181**	**28.0709**

Table 1. Quantitative evaluation of PSNR.



Warped View1 to View3 HHF Criminisi Proposed approach Ground truth

Fig. 4. Zoomed region from the Aloe dataset.

Thus, Ψ_q should be a patch that has similar texture and colors with Ψ_p.

4. EXPERIMENTAL RESULTS

To evaluate the proposed approach we use the datasets and disparity ground truth from the well known Middlebury dataset [12]. We also compare our results with the traditional exemplar-based inpainting approach [6] and the Hierarchical Hole-Filling (HHF) from Solh and AlRegib [11] qualitatively, through visual inspection, and quantitatively, using the Peak Signal-to-Noise Ratio (PSNR). It is important to note that the boundaries of the interpolated images (either left or right, depending on the reference image used) do not contain any valid information. Although our approach, as well as [6] and [11] are able to fill out those regions, they extrapolate image information. These portions should be visually coherent, but they are not taken into account when evaluating the PSNR.

Results for the proposed method and competitive approaches are shown in Table 1. For the tests we used the ground truth disparity from views 1 and 5, putting the synthetic camera in the location of view 3. As it can be observed, our method outperforms both approaches with respect to the PSNR metric for all tested datasets. Figure 4 shows a cropped and zoomed region of the Aloe dataset. Visual inspection indicates that the use of the disparity information to guide the hole filling algorithm was able to correctly propagate the background information within the holes. In contrast, the results from HHF are much blurrier, which is a expected result from the used multi-resolution approach that fills the hole with a combination from all the surrounding pixels, regardless of their disparity. More results are available at `http://www.inf.ufrgs.br/~aqoliveira/research/`.

5. CONCLUSION

In this work we first propose a simple solution for the removal of both ghosts and cracks, common visual artifacts in DIBR view interpolation techniques. For the cracks problem, we first classify crack regions using simple morphological operators, followed by the fast inpaint algorithm proposed by Oliveira et al. [4]. And to eliminate the ghost artifacts we identify the possible ghost regions and move them if necessary. By eliminating the ghosts we also help the hole filling algorithm, since they will not propagate those artifacts to the hole.

For the hole filling problem we propose to extend the work from Criminisi and collaborators [6] by changing the hole filling order using the depth information. By enforcing the texture propagation from the background to the foreground we were able to outperform the hole filling algorithm from Solh and AlRegib [11] in most of the tests, and obtain a much more sharp interpolated view.

As future work, we would like to investigate other disparity-based penalty metrics. We also intend to explore the disparity within the to-be-inpainted patches to reduce the search area for good patches, thus reducing the computational cost. Another possible path for future work is the extension of the patch selection scheme for video view interpolation, in which temporal coherence is an additional constraint.

Acknowledgments

We gratefully acknowledge the partial financial support from CAPES through grant BRANETEC 013/2013 and CNPq through grant 478730/2012-8.

6. REFERENCES

[1] Kwan-Jung Oh, Sehoon Yea, and Yo-Sung Ho, "Hole filling method using depth based in-painting for view synthesis in free viewpoint television and 3-d video," in *Picture Coding Symposium, 2009. PCS 2009*, 2009, pp. 1–4.

[2] Y. Mori, N. Fukushima, T. Fujii, and M. Tanimoto, "View generation with 3d warping using depth information for ftv," in *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, 2008*, 2008, pp. 229–232.

[3] Karsten Mller, Aljoscha Smolic, Kristina Dix, Philipp Merkle, Peter Kauff, and Thomas Wiegand, "View synthesis for advanced 3d video systems.," *EURASIP J. Image and Video Processing*, vol. 2008, 2008.

[4] Manuel M. Oliveira, Brian Bowen, Richard McKenna, and Yu sung Chang, "Fast digital image inpainting," in *PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON VISUALIZATION, IMAGING AND IMAGE PROCESSING*. 2001, pp. 261–266, ACTA Press.

[5] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester, "Image inpainting," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, New York, NY, USA, 2000, SIGGRAPH '00, pp. 417–424, ACM Press/Addison-Wesley Publishing Co.

[6] Antonio Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *Image Processing, IEEE Transactions on*, vol. 13, no. 9, pp. 1200–1212, 2004.

[7] I. Daribo and H. Saito, "A novel inpainting-based layered depth video for 3dtv," *Broadcasting, IEEE Transactions on*, vol. 57, no. 2, pp. 533–541, 2011.

[8] Alexandru Telea, "An image inpainting technique based on the fast marching method," *Journal of Graphics Tools*, vol. 9, no. 1, pp. 23–34, 2004.

[9] Alexandre Hervieu, Nicolas Papadakis, Aurélie Bugeau, Pau Gargallo, and Vicent Caselles, "Stereoscopic image inpainting: distinct depth maps and images inpainting," in *IEEE International Conference on Pattern Recognition*. IEEE, 2010, pp. 4101–4104.

[10] Yu Mao, Gene Cheung, Antonio Ortega, and Yusheng Ji, "Expansion hole filling in depth-image-based rendering using graph-based interpolation.," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 1859–1863.

[11] M. Solh and G. AlRegib, "Hierarchical hole-filling for depth-based view synthesis in ftv and 3d video," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 6, no. 5, pp. 495–504, 2012.

[12] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light," in *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, 2003, vol. 1, pp. I–195–I–202 vol.1.