IMPROVING THE QUALITY OF DEPTH IMAGE BASED RENDERING FOR 3D VIDEO SYSTEMS

Zefeng Ni[*], *Dong Tian*[†], *Sitaram Bhagavathy*[†], *Joan Llach*[†], *B. S. Manjunath*[*]

[*]Department of Electrical and Computer Engineering, University of California, Santa Barbara
[†]Thomson Corporate Research, Princeton, NJ, USA

ABSTRACT

In 3D Video (3DV) applications, a reduced number of views plus depth maps are transmitted or stored. When there is a need to render virtual views in between the actual views, the technique of depth image based rendering (DIBR) can be used to generate the intermediate views. To address the problem of noisy depth information in 3DV systems, we propose novel methods that can be easily incorporated into DIBR to improve synthesized image quality. These include: (1) a heuristic scheme with adaptive spatting that blends multiple warped reference pixels based on their depth, warped pixel positions and camera parameters; (2) an approximation of the first scheme with up-sampling for fast processing; (3) boundary only splatting; and (4) view weighting based on hole distribution. Experiment results show that the proposed methods can improve synthesis quality significantly.

Index Terms— View synthesis, view interpolation, depth image based rendering, 3DV.

1. INTRODUCTION

There is an increasing interest in 3D video (3DV) systems as the technology advances rapidly in 3D scene capturing, processing, transmission and displaying. For example, 3D display in home environment without the wearing of specific glasses has become possible [1]. However, simultaneous use of many views significantly increases the amount of raw data compared with 2D video or stereo-pair video. One efficient method to reduce data rate is to use a multiview video plus depth (MVD) format, i.e. only a subset of views (reference views) from the N display views are transmitted to the receiver. For each reference view, its corresponding depth map and meta data such as camera parameters are conveyed together with the video signal. All the other views are then synthesized using the technique of depth image based rendering (DIBR). Fig. 1 shows such a framework of 3DV systems.

The success of 3DV systems in Fig. 1 depends a lot on the quality of view synthesis at the receiver. However, high-quality synthesis with DIBR is a challenging task especially when the depth map is noisy and no extra scene information such as 3D surface property is known. This is because per-pixel depth for 3DV systems is often estimated with passive computer vision techniques such as stereo rather than generated from laser range scanning or computer graphics model. In addition, the quantization of pixel depth also introduces depth noise at the receiving end.

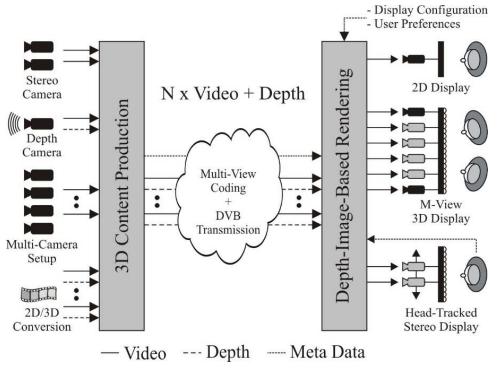


Fig. 1. An example of 3DV system with MVD format [2]

DIBR is a technique of view synthesis which uses images captured from multiple calibrated cameras and their associated per-pixel depth information. Conceptually, this method can be understood as a two-step process: (1) 3D image warping: it uses depth data and associated camera parameters to back-project pixel samples from reference images to the proper 3D locations and re-project them onto the new synthesized image space [3]; and (2) reconstruction and re-sampling: determination of pixel sample values in the synthesized image.

Given per-pixel depth information and camera parameters, it is straightforward to warp reference pixels onto the synthesized views. The difficult problem is how to estimate pixel value in the target view from its surrounding warped reference view pixels, i.e. the re-sampling (view blending) problem. Fig. 2 illustrates this basic problem. The synthesis method can be pixel-based (splatting) [4] or (triangular) mesh-based [5, 6]. For real-time processing in 3DV, pixel-based methods are often favored to avoid complex and computationally expensive mesh generation. The question is: to

This paper is supported in part by a grant from ONR #N00014-07-1-0029.

estimate the pixel values in the target view, how to utilize surrounding warped pixels around the target pixel? In this paper, we propose four novel methods that can be easily incorporated into DIBR to improve synthesized image quality. They are (1) a heuristic scheme with adaptive spatting that blends multiple warped reference pixels based on their depth, warped pixel positions and camera parameters; (2) an approximation of the first scheme with up-sampling for fast processing; (3) boundary only splatting; and (4) view weighting based on hole distribution. Experiments show that the proposed simple heuristics can improve synthesis quality significantly.

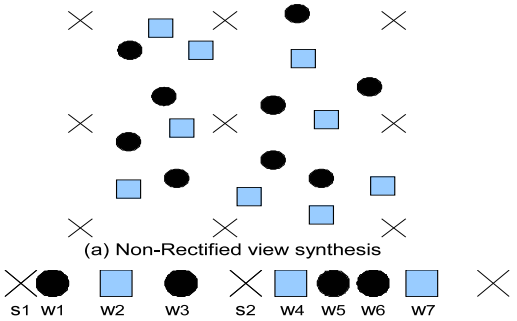


Fig. 2. Pixel resampling ("X": pixels on target view; "circle, squares": pixels warped from reference views, shapes denote different reference views)

2. PROBLEM STATEMENT

Ideally, we would want to achieve the view synthesis in a theoretically-optimized fashion, as in [7, 8] for multiview image reconstruction. However, it is hard to extend the homography-based analysis in [7, 8] to the case of depth-based view warping. In addition, the computational cost of the algorithms in [7, 8] make them unsuitable for real-time operation in 3DV systems.

An alternative method is to do it in an ad-hoc fashion. A simple method is to round the warped samples to its nearest pixel location in the destination view as in [9, 10]. When multiple pixels are rounded to the same location in the synthesized view, Z-buffering [4] is a de-facto solution, i.e. choosing the front-most pixel. This strategy, however, can often result in pinholes in any surface that is slightly under-sampled, especially along object boundaries. The most common solution is to map one pixel in the reference view to several pixels in the target view, a process known as *splatting*.

If a reference pixel is mapped onto multiple surrounding target pixels in the target view, most of the pinholes can be eliminated. However, some image detail might be lost, i.e. there is a trade-off between pinhole elimination and loss of details. When multiple warped pixels (candidate pixels) are mapped to the target pixel, shall we simply apply Z-buffering to pick one or try to interpolate them with some kind of weighting? Note that the problem should be addressed in a way that is robust to the noisy depth information.

3. PROPOSED METHODS

Existing splatting methods[4, 11, 12, 13] have achieved good results. However, they are designed to work with high precision depth and might not be adequate for low quality depth. In addition, there are aspects that many algorithms take for granted (e.g. from a graphics model), such as per-pixel surface normal or 3D point-cloud, which is not available in 3DV. For example, a common method of splatting is to map each warped pixel to its neighboring target-view pixels and use Z-buffering when multiple warped pixels are mapped to the same target pixel. This method (**Method 0**) works well if the depth value are perfect. However, its synthesized image quality is very prone to noisy depth input. Here we propose some novel methods to improve the synthesis result.

3.1. Method 1: Heuristic blending with adaptive splatting

The idea of using heuristics to improve blending is not new. For example, in [13], when blending two warped pixels from different reference views, their depth values are first compared. If the depth difference is within a given threshold ϵ, their color values are linearly interpolated with weights based on cameras' positions. Otherwise, the color of the pixel closer to the camera (i.e. Z-buffering) is set as the target pixel color. This threshold ϵ provides some tolerance to noisy depth. We extend the idea further by blending warped reference pixels based on their depth level, warped pixel positions and camera parameters. For simplification, we use rectified view synthesis as an example, i.e. estimate the target pixel value from the candidate pixels on the same horizontal line (Fig. 2b). Fig. 3 shows the proposed heuristic view blending method, which uses essentially an adaptive splatting scheme.

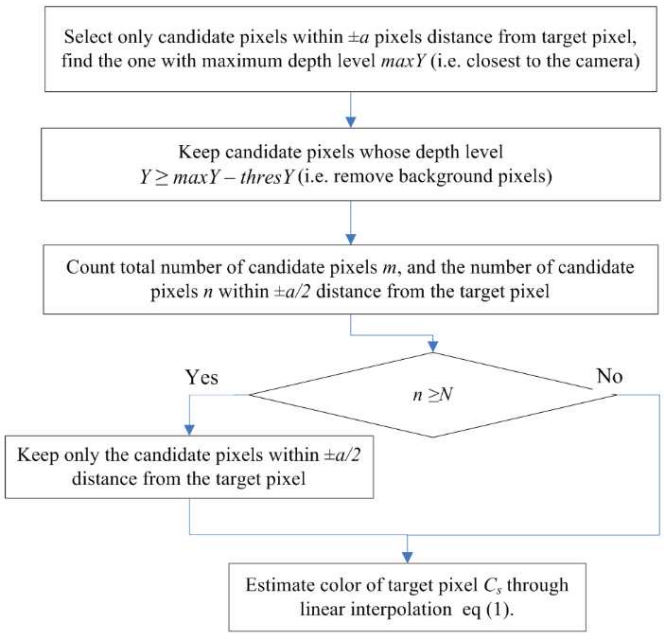


Fig. 3. Heuristic view blending for rectified views

For each target pixel, warped pixels within $\pm a$ pixels distance from its position on the image are chosen as candidate pixels. The one with maximum depth level $maxY$[1] is found. Parameter a here is crucial. If it is too small, pinholes will appear. If it is too large, image details will be lost. It can be adjusted based on some prior knowledge about the scene or the depth precision. In our experiments, $a = 1$ works most of time for depth estimated with passive vision based approach.

With Z-buffering, the candidate with maximum depth level will determine the pixel value at the target position. Here, we propose to keep the other candidate pixels as long as their depth levels $Y$are close to the maximum depth, i.e., $(Y \geq maxY - thresY)$, where $thresY$ is a threshold parameter. In our experiments, $thresY$ is set to 10 for 8 bits depth quantization. It could vary according to some prior knowledge of the scene. Let us denote by m the number of candidate pixels found so far.

To keep image details, if there are "enough" number of candidates within $\pm a/2$ pixels distance from the target pixel, only these candidates will be used to estimate the target pixel color. Let us define the number of such candidate pixels as n. If $n \geq N$, i.e., if n is larger than a preset threshold N (we set it to 4 when $thresY = 10$ and there are two reference views), then only these n candidates are used. This process of choosing warped pixels closer to the target pixel can be repeated further if a is large.

After n_p candidate pixels are selected, the next task is to estimate the target pixel value C_s. Let us define the value of a candidate pixel i to be C_i, which is warped from reference view r_i and whose corresponding image distance to the target pixel is d_i. We find that the following linear interpolation works very well,

$$C_s = (\sum_{i=1}^{n_p} w_i C_i) / \sum_{i=1}^{n_p} w_i, \quad (1)$$

with $w_i = (a - d_i)W(r_i, i)$ and $W(r_i, i)$ is the weight factor assigned to different views. It can be simply set to 1. For rectified views, we set it based on baseline spacing l_i (the camera distance between view r_i and the target view), e.g. $W(r_i, i) = 1/l_i$.

The proposed scheme in Fig. 3 essentially selects/blends candidate pixels from surrounding warped reference pixels based on their depth levels, their warped image positions and camera positions. It is easily extended to the case of non-rectified views. The only difference is that candidate pixels will not be on the same line of the target pixel (Figure 2a). But the same principle to select/blend candidate pixels based on their depth and their distance to the target pixel can be applied. For more precise weighting, $W(r_i, i)$ can be further determined at pixel level, e.g. using the angle determined by the point in 3D and cameras' positions.

[1]Depth level here refers to quantized depth value, e.g 0-255. The larger the value is, the closer it is to camera.

3.2. Method 2: Approximation with up-sampling

Method 1 might appear to be too complicated when comparing with method 0. Here we propose a scheme similar to method 0 but approximates method 1 with some compromise of synthesis quality. In the case of rectified views, we first insert a new target pixel at all half-pixel positions in the target view, i.e. up-sampling along the horizontal direction. Then for each target pixel, a simple splatting and Z-buffering (as in method 0) is applied to estimate its value. This is equivalent to setting $thresY = 0$ in the method 1. To generate the final synthesized view, a simple down-sampling filter (e.g., $\{1, 2, 1\}$ works fine in our experiments) is used. This filter approximates the weight w_i in eq. (1). This approximation with up-sampling allows efficient implementation, especially for embedded systems. Of course, the same approach can also be applied for non-rectified views. The only difference is that we have to up-sample the image along both horizontal and vertical directions.

3.3. Method 3: Boundary-only splatting

As explained above, splatting is used to reduce pinholes, i.e. a warped pixel is mapped to multiple neighboring target pixels. For example,in the case of rectified views (Fig. 2b), warped pixel $W1$ is mapped to target pixels $S1$ and $S2$ (i.e. $W1$ is a candidate pixel for estimation of both $S1$ and $S2$). However, we find this could affect the image quality (i.e. image details are lost due to splatting) especially when sub-pixel precision (e.g. with Method 2) is used. Noticing that pin-holes mostly occur around the boundary between foreground and background, i.e. boundary with large depth discontinuity, we propose to apply splatting only for pixels close to the boundary. In the case of Fig. 2b, if pixel $W1$ is not close to boundary (e.g. further than 50 pixel distance from the boundary), it is mapped only to its closest target pixel $S1$. Note that "boundary" here only refers to the part of the image with large depth discontinuity and hence is easy to detect from the depth maps.

3.4. Method 4: View weighting based on hole distribution

For DIBR implementation, the blending is often achieved through two steps: synthesize a virtual image from each reference view separately (with any method above) and then merge all synthesized images together. For each synthesized image (from a single reference view), some pixels in the synthesized image are never assigned a value during the blending step. These locations are called holes, often caused by dis-occlusions (previous invisible scene points in the reference views are uncovered in the synthesized view due to differences in viewpoint) or due to input depth error. In the case of two reference views, we have two synthesized images. Let us define the number of hole pixels around pixel i is $holeCount1$ and $holeCount2$ respectively in each synthesized image. If they differ a lot (e.g. $|holeCount1 - holeCount2| \geq 2$ in our experiments), we propose to ignore the one (say view r_i) with more holes

around it, i.e. set its $W(r_i, i)$ to 0 at pixel i. Otherwise, normal interpolation (e.g. eq(1)) holds. The rationale is that "for the area around pixel i, candidate pixels from view r_i are less reliable probably due to much larger depth noise."[2]

4. EXPERIMENTAL RESULTS

The proposed heuristic methods have been tested extensively in experiments. Compared with methods using simple splatting and Z-buffering (method 0), the proposed methods could improve the synthesis quality significantly, especially for method 1 with adaptive splatting. Due to the space limit, we illustrate the result with only the "door flower" sequence. The depth maps used in our experiments are estimated using the 3DV reference software [10]. Results with four different synthesis cases are shown here, i.e. synthesis view 7 and 8 from view 10 and view 5, and synthesis view 7 and 8 from view 9 and view 6. The synthesis images are compared with original images both visually (Fig. 4) and objectively (Fig. 5). We can thus see the superiority of the proposed methods over method 0. For example (Fig. 4), the vertical lines are preserved much better with proposed methods. Observe that method 1 increases PSNR by over 1dB on average over method 0 (Fig. 5).

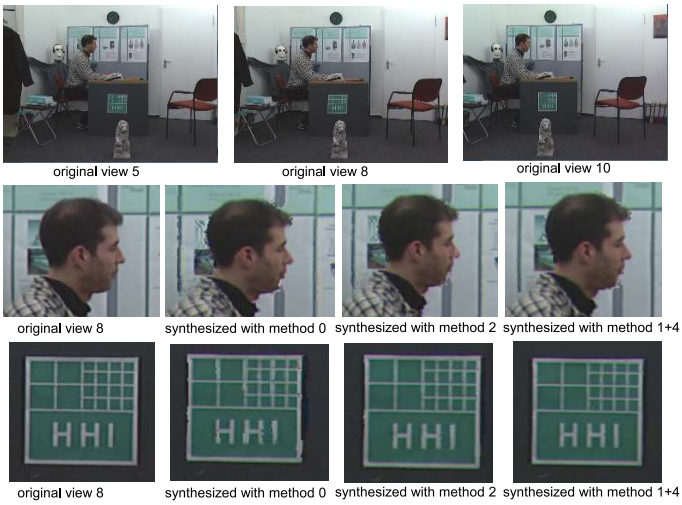
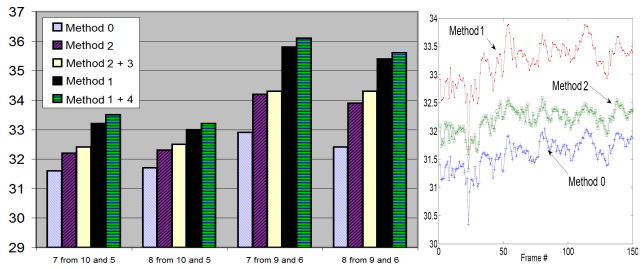


Fig. 4. Sample frames (synthesize view 8 from view 5 and 10 in the "door flower" sequence).

5. CONCLUSIONS

In 3DV systems, depth information estimated with passive vision methods are typically very noisy. This noisy depth could severely affect image synthesis quality using simple depth-image based rendering methods. In this paper, we proposed novel methods that can be easily incorporated into DIBR to improve synthesized image quality, including a heuristic scheme with adaptive spatting that blends multiple warped reference pixels based on their depth, warped pixel positions and camera parameters, its approximation with up-sampling for fast processing, boundary only splatting and view weighting based on hole distribution. The proposed methods do not require any extra prior scene information. Experiments show the effectiveness of the proposed methods.

[2]This is very likely when the input depth maps for different views are estimated separately with stereo images based on disparity estimation.



(a) (b)

Fig. 5. PSNR of synthesized videos (compared with original image). (a) Average PSNR (b) PSNR at each frame (synthesize view 7 from view 5 and 10)

6. REFERENCES

[1] J. Konrad and M. Halle, "3-D displays and signal processing," *IEEE Signal Processing Magazine*, vol. 24, no. 6, pp. 97–111, 2007.

[2] MPEG Video Subgroup, "Applications and requirements on FTV," *MPEG output document w9466*, October, 2007.

[3] C. Fehn, "Depth-image-based rendering (DIBR), compression and transmission for a new approach on 3D-TV," in *Proceedings of SPIE Stereoscopic Displays and Virtual reality Systems XI*, 2004, pp. 93–104.

[4] W. Mark and G. Bishop, "Efficient reconstruction for post-rendering 3D image warping," Tech. Rep., Univ. of Northern Carolina Chapel Hill, 1998.

[5] T. Kanade, R. W. Rander, and P. J. Narayanan, "Virtualized reality: constructing virtual worlds from real scenes," *IEEE Multimedia Magazine*, vol. 1, no. 1, 1997.

[6] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," in *Proceeding of SIGGRAPH Conference*, 2004, pp. 600–608.

[7] L. Wang, S. B. Kang, R. Szeliski, and H.-Y Shum, "Optimal texture map reconstruction from multiple views," in *Computer Society Conference on Computer Vision and Pattern Recognition*, 2001, pp. 347–354.

[8] J. Domke and Y. Aloimonos, "Multiple view image reconstruction: A harmonic approach," in *Computer Society Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[9] D. Tian, P. Pandit, and C. Gomila, "Simple view synthesis," *MPEG document M15696*, 2008.

[10] M. Tanimoto, T. Fujii, K. Suzuki, N. Fukushima, and Y. Mori, "Reference software for depth estimation and view synthesis," *MPEG document M15377*, 2008.

[11] M. M. Oliveira, G. Bishop, and D. McAllister, "Relief texture mapping," in *Proceedings of SIGGRAPH Conference*, 2000, pp. 359–368.

[12] P. M. Zwicker, J. van Baar, and M. Gross, "Surface splatting," in *Proceedings of SIGGRAPH Conference*, 2001, pp. 371–378.

[13] A. Smolic, 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," in *IEEE Int. Conf. Image Processing*, 2008, pp. 2448–2451.