Improved depth-image-based rendering algorithm

Article · May 2011

DOI: 10.1109/3DTV.2011.5877183

CITATIONS
2

READS
48

4 authors, including:

Mårten Sjöström
Mid Sweden University
108 PUBLICATIONS **855** CITATIONS
SEE PROFILE

Roger Olsson
Mid Sweden University
63 PUBLICATIONS **509** CITATIONS
SEE PROFILE

Sjöström, M.; Härdling, P.; Karlsson, L.S.; Olsson, R., "Improved depth-image-based rendering algorithm," *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*,16-18 May 2011.

http://dx.doi.org/10.1109/3DTV.2011.5877183

IMPROVED DEPTH-IMAGE-BASED RENDERING ALGORITHM

Mårten Sjöström, Peter Härdling, Linda S. Karlsson, Roger Olsson

Mid Sweden University, SE-85170 Sundsvall, Sweden

ABSTRACT

The present work analyses a layered depth-image-based-rendering algorithm based on possible errors occurring with perspective 3D warping. The outcome is improvements to the algorithm that treats depth reliably for scenes containing several levels of foreground objects. The filling of holes of different kinds is addressed so that results have better visual quality. The analysis compares the results of the algorithm with a reference algorithm for the potential error types, and visual examples exhibit the consequences of the improvements. Different objective metrics give ambiguous results, which may be explained by the reduction of structure caused by the reference algorithm.

Index Terms— 2DD, 3D, DIBR, Multiview plus Depth, SLERP, warp

1. INTRODUCTION

In production of 3D films for multiview displays, all pixels of each perspective view may be computed perfectly when starting from a computer generated scene. In the case of image-based rendering starting from recorded video sequences, certain typical discrepancies will occur in the perspective views. Having these typical discrepancies as starting points, it is possible to improve results.

The format 2D plus Depth (2DD) depicts a scene by a two-dimensional image supplemented by a depth value for each pixel. The depth information may be used to render virtual views of the scene for the purpose of free-viewpoint or multiview displays. A drawback with the 2DD format is the occluded areas that occur when foreground objects have been moved relative to the background. In order to allow a wider range of virtual views, and to omit occlusions, several 2DD sequences may be used in parallel. This Multiview plus Depth (MVD) format permits virtual views to be rendered from two adjacent 2DD sequences so that occluded areas from one sequence can be filled by the other sequence. However, certain rendering problems still arise with the MVD [1]: *cracks* due to integer round-offs are exhibited where the image has been stretched; *empty regions* show up where information cannot be obtained from the 2DD sequences; *corona-like effects* appear around foreground objects due to erroneous depth information; and *unnatural contours* may occur in the transformed virtual views due to rounding errors and different lighting conditions in the two original images.

The present work presents an improved algorithm and an analysis with the mentioned problems as starting points. The algorithm is based on the work by Zitnick *et al*. [2] that has been further improved by Müller *et al*. [1]. We have modified the algorithm in order to deal with scenes containing several levels of foreground objects; we have also extended it to manage different types of holes, with an improved appearance as a result. The improved algorithm has been evaluated by comparing it to a fundamental algorithm based on perspective 3D warping [3, 4] with additional post-processing [5] by using objective metrics and visual inspection.

The paper is outlined as follows. Section 2 explains the fundamentals of 3D warping, the algorithm by [1] and the improvements in the proposed algorithm. The experimental set-up, the reference algorithm and the evaluation criteria are described in Section 3. The results and analysis follows in Section 4, and the conclusions are found in Section 5.

2. 3D WARPING ALGORITHM

The depth-image-based rendering algorithms in the present article are based on perspective 3D warping. The 2DD information and the camera parameters are then required to produce a virtual view. The extrinsic parameters express the necessary rotation **R** and translation **t** to transform a 3D point with homogeneous coordinates $\mathbf{M} = (x, y, z, 1)^T$ in the scene into a camera pixel $\mathbf{m} = (u/z, v/z, 1)^T$. The intrinsic parameters are represented by the calibration matrix **K** that describes the focal distance, image centre and camera pixels sizes. Together they define the projection matrix $\mathbf{P} = \mathbf{KI}[\mathbf{R} \mid \mathbf{t}]$. The forward warping produces a new pixel in the virtual view $z_v \mathbf{m}_v = z_L \mathbf{P}_v \mathbf{P}_L^{-1} \mathbf{m}_L$, where subscripts *v* and *L* corresponds to virtual and left views, respectively. In the same way, pixels of the 2DD on the right-hand side produce new pixels in the virtual view. A detailed description is found in [3, 4].

2.1. Layered algorithm

The algorithm proposed in [3] divides the original views into layers depending on the confidence of the information.

978-1-61284-162-5/11/$26.00 2011 © IEEE

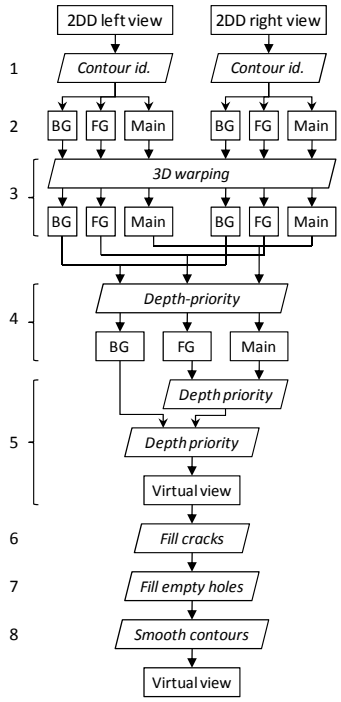

Fig. 1 The eight processing steps of the layered algorithm

More confident layers are given priority in the selection of pixel information. A detailed description is found in [1].

1. Contours are identified in left and right view based on respective depth information.
2. Areas in a vicinity of the contours are divided into foreground and background layers. Remaining parts of the view is assigned to the confident main layer.
3. The layers are transformed into a new common virtual view, but are held separate in different images. The pixel closest to the viewer is selected in case of several pixels are projected to the same image position.
4. The three layers from left and right views are merged into corresponding three common layers. The pixel closest to the viewer is selected if several pixels are projected to the same image position. In case the difference in depth value is below a certain limit, information from both views is merged by weighting with the distance to the original view.
5. The three layers are merged into a common view in two steps. Firstly, the foreground and main layers are combined by taking the information closest to the viewer. Secondly, information is added from the background layer for pixels where information is missing.
6. Cracks where the background erroneously seeps through are identified and replaced by interpolated values if the difference in depth surpassed a certain limit.
7. Remaining (larger) empty regions are filled by background information in its horizontal vicinity.
8. New contours around foreground objects are identified and smoothed by low pass filtering.

Fig. 1 depicts the algorithm steps.

2.2. Improvements

We have supplemented the described algorithm in order to improve its performance. The following steps have been altered:

In step 1, the sensitivity was increased in order to detect more edges so that objects between foreground and background could be discovered.

Step 2 has been supplemented with an adaptive limit for division between foreground and background layers in order to take into account the edges' position in depth: the limit is set to the minimal depth value in a 5x5 surrounding of each edge point with an extra margin of 2 out of 256 depth values.

We have introduced an intermediate step between 3 and 4, step "3.5", where empty cracks (identified by depth value = 0) are filled by the median of adjacent pixels. This reduces conspicuous crack fillings from another view with a different color tint due to a slightly different exposure.

Step 5 had to be adjusted because the higher sensitivity of the edge detector in step 1 results in edges *within* foreground objects. Foreground objects may then be partly assigned to the background layer with the possible consequence of faulty translucence of the background in the main layer. Therefore, step 5 was altered to take values from the background layer if it is closer to the viewer than the foreground and main layers. A margin of 2 out of 256 depth values specifies how much closer the background layer must be. If transitions between foreground and background objects are less sharp, this margin must be increased.

The limit for translucent cracks in step 6 was adjusted to 10 (originally 40) in order to find such cracks with less depth difference.

We have classified larger empty regions into three types in step 7 according to depth values in adjacent pixels because these need to be treated differently. Type 1 has foreground and background data on opposite sides and is due to smaller movements of foreground objects relative to background. Type 2 has background data on both sides due to larger movements. Type 3 has foreground data on both sides due to occluded background data in both left and right views. We have introduced a minimal difference of 15 out of 256 depth values to distinguish holes of type 1 from 2 and 3, computing the difference in depth on each side. In order to differentiate between holes of type 2 and 3, we have applied a fix limit for each sequence, where of 256/3 depth values worked well. Holes smaller than 3 pixels wide are always considered as type 2. Holes next to the image edge belong to type 1 where the depth of one side is considered background. Holes between both image edges are assigned to type 3.

Table 1 Rendering Errors

	Rendering errors	Reference algorithm	Improved algorithm
1	Empty cracks	Median filtering over whole image	Interpolation of missing pixels only
2	Translucent cracks	Median filtering over whole image	Search for cracks and replace erroneous pixels
3	Corona-like effects	Joints smoothed by median filtering	Reduced priority of background layer
4	Unnatural contours	Priority of next view at transformation, and leveling of edges and median filtering over the whole image.	Priority and mixing of foreground layers along with mean filtering along contours.
5	Empty regions	Linear interpolation in the rendered image. No usage of depth information.	Horizontal filling with constant value form background.
6	Erroneous depth value	--	--

Type 1 holes are filled horizontally with the median color of four pixels from the background side; type 2 holes are filled with a constant color interpolated from the median from each side of the hole; and type 3 holes are filled with a guessed background color. This guess equals the median color of the whole view. We have furthermore introduced a low-pass filtering of filled holes in order to reduce their striped appearance.

The contours in step 8 are identified by using a more sensitive edge detector in order to find all edges to smooth.

3. EXPERIMENTAL SET-UP

We have used the sequences *Balletdancer* and *Breakdancing* [2,6]. The improved algorithm has been evaluated according to the quality of rendered images along the spherical linear interpolation (SLERP) between left and right views. The way the algorithm treats cracks has especially been investigated.

As reference algorithm has been used a standard 3D warping algorithm that further smoothes the resulting view in a vicinity of errors by median filtering [3-5].

Both improved and reference algorithms have been implemented in MATLAB. The limit values given by [1] had therefore to be adjusted. The Canny edge detector provided in MATLAB was set to 0.05 (adjusted value), and the standard deviation of the Gauss filter to 1.

3.2. Evaluation criteria

The results have been evaluated by visual comparison and objective metrics. Each intermediate view in three frames of the sequences has been rendered as a virtual view, a total of six images per scene. The original views at the same positions have been used as references. The differences between rendered and original views were measured by using PSNR (peak signal-to-noise ratio) and MSSIM (mean structural similarity) [7]. For each rendered virtual view, the number of missing pixels was identified as it gives a scale to the portion of estimated pixels in the virtual view.

Fig. 2 Original view (left) and rendered view with errors

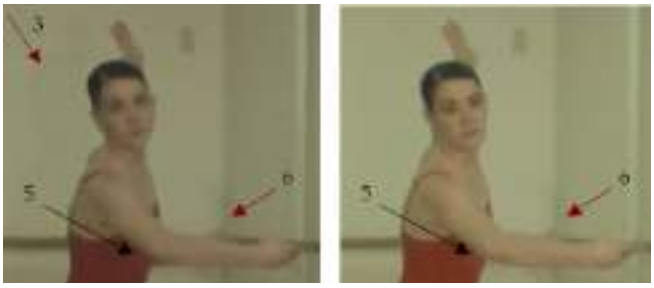

Fig. 3 Rendered views with reference algorithm (left) and improved algorithm

4. RESULTS AND ANALYSIS

The rendering errors and their treatment by both algorithms are given in Table 1. See also Fig. 2. Both algorithms manage to cover round off cracks well (error 1 and 2). See Fig. 3. The improved algorithm preserves better the image sharpness. Both algorithms also manage to remove much of the corona-like effects (error 3), but the corona can still be discern for the reference algorithm, which may be caused by differences in exposure between left and right views. The improved algorithm also deals better with smoothing unnatural contours (error 4, not in figure). Fig. 3 depicts that larger empty regions (error 5) are filled with a color consistent with the original, where the reference algorithm interpolates from pixels of each side. The reference algorithm does, however, have the better result when a full line is missing (usually at image top or bottom due to alignment problems). Both algorithms produce erroneous colors for empty regions in the background (type 1 holes). Neither of the algorithms manages to deal with errors due to erroneous depth values (error 6), which emphasizes the importance of a correct depth map.

Noteworthy is that the results of the reference algorithm are highly dependent on whether left or right view has been prioritized; the PSNR differs up to 2 dB in certain virtual views. The improved algorithm, on the other hand, generally produces resulting views with an even and higher PSNR. See Fig. 4. Surprisingly, a comparison using MSSIM shows consistently better results for the reference algorithm. A possible reason is that MSSIM rewards views with less

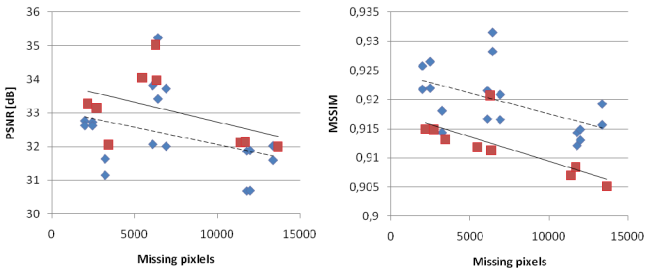

Fig. 4 PSNR and MSSIM vs. missing pixels for full algorithms of the Balletdancer: improved algorithm (□) and reference algorithm (◊).

(erroneous) details, which is obtained by the median filtering. Both algorithms are dependent on the number of missing pixels (sizes of the holes), which is manifested by the linear approximations in Fig. 4. The *breakdancer* sequence gives similar results although with a larger spread especially among the PSNR values, which may be caused by its larger portion of erroneous depth values.

More specific consequences to the improvement introduced in the present work can be seen in Fig. 5 and 6. The former depicts the results of an increased sensitivity to the Canny edge detector, where foreground objects closer to the background needed to be identified. Undesired corona-like effects appeared when a large Canny threshold (0.43) is used. The adjusted value (0.05) avoided the effect. Fig. 6 demonstrates the result of the improvements in step 5. The new Canny threshold may result in a foreground object being divided into both foreground and background layers. This background layer then obtains a lower priority to the benefit of the main layer of an object farther into the view. See centre image of Fig. 6. The improved step 5 verifies the depth values so that the objects mutual placements in depth are respected. The filling color for type 3 holes in step 7 has been selected to the median color of the whole view. This works well as long as most pixels belong to the background, as in the used sequences.

Thresholds in this investigation may need to be adjusted for other types of images, especially of different size.

5. CONCLUSIONS

An improvement to a layered depth-image-based-rendering algorithm has been presented. Both the improvements and the analysis are based on the different types of errors that occur in perspective 3D warping. In particular, scenes containing several levels of foreground objects are treated in a depth-consistent way. The improved algorithm also manages different types of holes, with an improved visual result. A comparison with a fundamental perspective 3D warping algorithm demonstrates the improvements along with visual examples. The improved algorithm scores better using PSNR, but not with MSSIM, which may be caused by a reduction in structure due to the median filter in the fundamental algorithm.

Fig. 5 A large Canny threshold may imply corona-like effects (lower), whereas a more sensitive edge detection avoids those (upper).

Fig. 6 Improvements to step 5 implies a correct placement in depth.

6. ACKNOWLEDGEMENTS

This work is supported by the EU Objective 2 - project “Digital 3D Signage”, and the KK-foundation project “3D video”.

7. REFERENCES

[1] K. Müller, *et al*., “View Synthesis for Advanced 3D Video Systems,” *EURASIP Journal on Image and Video Processing*, 2008.

[2] C.L. Zitnick, *et al*., “High-quality video view interpolation using a layered representation,” *ACM Trans. on Graphics*, vol. 23, pp. 600-608, 2004.

[3] C. Fehn, “Depth-image-based rendering (DIBR), compression and transmission for a new approach on 3D-TV,” In *Stereoscopic Displays and Virtual Reality Systems XI*, SPIE, pp. 93-104, 2004.

[4] Schreer *et al*., “3D Video communication: Algorithms, concepts and real-time systems in human centered communication,” John Wiley & Sons Ltd., 2005.

[5] E. Cooke *et al*., “Multiview synthesis: A novel view creation approach for free viewpoint video,” *Signal Processing: Image Communication*, vol. 21, pp. 476-492, 2006.

[6] S.B. Kang *et al*., “Projection test and results for Microsoft Research3D Video (Breakdancing frame),” *Microsoft Research*, Redmond, 2004.

[7] Z. Wang *et al*., “Image Quality Assessment: From Error Visibility to Structural Similarity,” *IEEE Trans. on Image Proc*., vol. 13, no. 4, pp. 600-612, 2004.