Depth Map Up-sampling Based on Edge Layers

Danillo Bracco Graziosi, Dong Tian*, Anthony Vetro
Mitsubishi Electric Research Labs, Cambridge, USA
* E-mail: tian@merl.com Tel: +1-617-621-7556

***Abstract*—Depth map images are characterized by large homogeneous areas and strong edges. It has been observed that efficient compression of the depth map is achieved by applying a down-sampling operation prior to encoding. However, since high resolution depth maps are typically required for view synthesis, an up-sampling method that is able to recover the loss of information is needed within this framework. In this paper, an up-sampling algorithm that recovers the high frequency content of depth maps using a novel edge layer concept is proposed. This algorithm includes a method for extracting edge layers from the corresponding texture images, which are then used as part of a non-linear interpolation filter for depth map up-sampling. In the present work, the up-sampling is applied as a post-processing operation to generate multiview output for display. Views synthesized with our up-sampled depth maps show the efficiency of our proposed technique relative to conventional interpolation filters.**

I. Introduction

Following up on the successful deployment of stereoscopic services for broadcast and packaged media based on the well-established AVC standard [1], there is growing interest in depth-based formats to support auto-stereoscopic displays and advanced free-viewpoint functionality [2]. New coding formats that include support for depth images are currently being standardized as extensions of both AVC and the emerging HEVC standard.

In the framework under consideration, a scene is captured with multiple cameras, and for each view, the corresponding depth map is also obtained. Depth values can be directly acquired with depth-range cameras or estimated by means of stereo correspondence algorithm. This information enables high quality depth-image based rendering (DIBR) algorithms to synthesize the scene from different viewpoints than the original camera views.

Considering the compression of this data format, comprised of multiple texture videos and their corresponding depth maps, it has been found that encoding a reduced resolution depth can reduce the bit rate substantially. However, this loss of resolution in the depth map could also adversely impact the quality of the rendering result. Conventional resampling techniques apply low-pass filters and linear interpolation filters to recover the lost information and reduce the quality degradation. Since the majority of the loss is in the high frequency, edges become smeared or blurred. This is especially problematic for DIBR since the edges represent depth discontinuities and errors on the depth edges correspond to shifts in the rendering error, which result in artifacts on the rendered object boundaries.

In order to better maintain edge sharpness, Oh et al. proposed a reconstruction filter based on the occurrence frequency of the pixel values inside a window [3] . Applied together with a median filter and a bilateral filter, the sharp edges of the depth maps were able to be reconstructed. Nevertheless, the reconstructed edge position might not match the original position, and synthesis artifacts could still be observed.

Another approach to restore the high frequency of interpolated depth maps is to exploit the correlation between texture and depth maps in the up-sampling process. This method exploits the fact that strong edges in depth maps that correspond to object boundaries in a scene are a subset of edges from the texture image, which can be used to restore the sharpness of the interpolated depth edges. For example, in [5] a conventional interpolation filter is used for depth map interpolation, but when the support filter crosses an edge position, the edge location prevents the interpolation filter from using values across that edge. Hence, the filter does not process pixels from different depth planes.

Edge detection is an important step in such approaches. The process may also be done implicitly, i.e., instead of using the edge locations, the interpolation algorithm might directly consider the texture values in the up-sampling procedure. In [4], the interpolation is done by choosing one out of four nearest neighbors based on pixel position and the texture value differences. Similarly, a trilateral filter was proposed in [6] and another related approach is described in [7]. However, in texture images, a mixture of background and foreground pixels is usually present in the transition between objects, i.e., the edges in texture may not be as sharp and localized as the edges in depth maps, which might affect the performance of algorithms that use texture for depth up-sampling.

One way to circumvent the problems of inaccurate texture edges is to work with the edges extracted from the original high resolution depth map. In [8], the authors propose the explicit transmission of edge locations to be used in their wavelet-based depth map coding approach. The use of accurate edges provides a high quality depth map coding, but the coding efficiency is offset by the need to send the edge information. Nevertheless, depth map edge information could be used also in texture coding, where higher bitrates savings could be achieved. An example of the use of depth to improve coding efficiency of texture was given in [9], where depth edges are used to enhance the intra prediction modes.

In this paper, we propose a novel method of depth map up-sampling, that can use either edge information extracted from texture or accurate depth map edge information that is explicitly coded and transmitted. The algorithm explores the geometrical similarities between edges present in the texture

and in the corresponding depth map and is based on the fact that depth values along the edges are homogeneous, while values across the edge orientation experience a sharp transition.

The proposed up-sampling method comprises a nearest-neighbor interpolation followed by a post-filtering procedure. The first step of the method is an edge selection process, which is described in Section II. Next, edge layers are created based on the selected edges as described in Section III. Section IV describes the final non-linear filtering step based on edge layer information. In Section V, simulation results are presented and discussed, and Section VI concludes the paper.

II. Edge Selection from Texture

The similarity between edges from texture and edges from depth map relies on their geometrical structures rather than their absolute position in the image, that is, depth map edges have a similar contour to edges in the texture, but they might not be in the exact same location. Also, most of the edges detected in depth map can also be found in texture, while many edges detected in texture are not present in depth maps, especially edges from texture within an object. In certain cases, an object outline might be clear in the depth map, but due to texture similarities between objects and their foreground, the corresponding edge cannot be detected in the texture. For these reasons, the use of texture edges to reconstruct depth edges is a non-trivial problem.

The following procedure is proposed for edge selection. First, the texture edges are extracted; a canny operator is used in our current implementation. Then, a morphological operation is performed on the low resolution depth map to obtain the area where depth maps edges should appear. An open-close operation is done with the low resolution depth map, and areas that are greater than a pre-defined threshold are considered as candidate edge locations. Next, only texture edges in the detected area of the corresponding low resolution depth map image are considered. The procedure to select edges ($\hat{e}$) from an edge map ($e(x,y)$) is described as follows:

$$\hat{e}(x,y) = e(x,y) \times N(x,y) \tag{1}$$

where

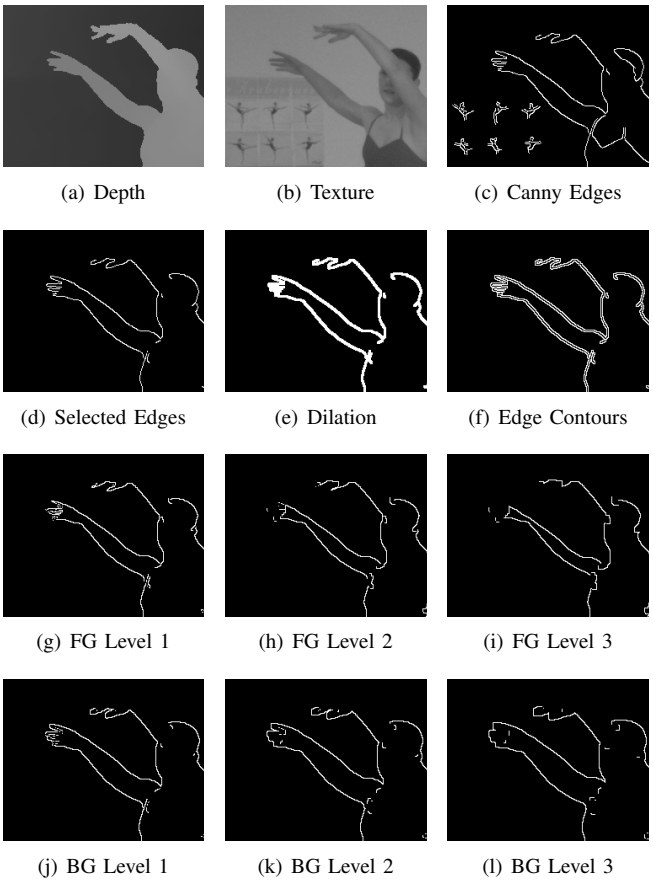
$$N(x,y) = \begin{cases} 1 & \text{if } M(x,y) \geq T \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$M(x,y) = \max(D(\frac{x}{s} \pm 1, \frac{y}{s} \pm 1)) - \min(D(\frac{x}{s} \pm 1, \frac{y}{s} \pm 1)) \tag{3}$$

where $D(x,y)$ is the corresponding low resolution depth image scaled by a factor of s, and T is the threshold value, which was set to 5 in our simulations. The edges and selected edges using the above process for a cropped region of the Ballet sequence are shown in Figs. 1(c) and 1(d), respectively.

III. Edge Contour Extraction

The object contours along the edge lines are obtained in the following manner. First, the selected edges from texture are dilated [1], resulting in an enlarged edge as shown in Fig. 1(e), then edges are extracted, resulting in two curves symmetrically opposite to each other along the edge line as shown in Fig. 1(f). The two lines are currently classified into either background contours or foreground contours according to the depth values perpendicular to the tangent of the edge curve, although different levels of classification could be considered.



(a) Depth (b) Texture (c) Canny Edges

(d) Selected Edges (e) Dilation (f) Edge Contours

(g) FG Level 1 (h) FG Level 2 (i) FG Level 3

(j) BG Level 1 (k) BG Level 2 (l) BG Level 3

Fig. 1. Details of the process of layer extraction.

Next, for each foreground or background line, a process of dilation and removal of the original line is done recursively, creating layers of pixels that follow the edge direction with different offsets to the depth discontinuity. Figs 1(g), 1(h) and 1(i) show the selection of object contours that are located in the foreground, while Figs. 1(k), 1(l) and 1(j) show the same procedure for contour lines located outside the object. The number of layer constructed is proportional to the down-scaling factor.

IV. Non-linear Edge Layer Filtering

The geometrical property of edges indicates that depth edge values should be homogeneous along the edge. So, once the edge contours are identified, a non-linear operation is performed on the depth samples along those contours in order to make their values more homogeneous, i.e., a smoothing is applies to pixels along the edge contours. Pixels that do not belong to the identified edge layers are not filtered, and a simple nearest neighbor interpolation is used for those areas.

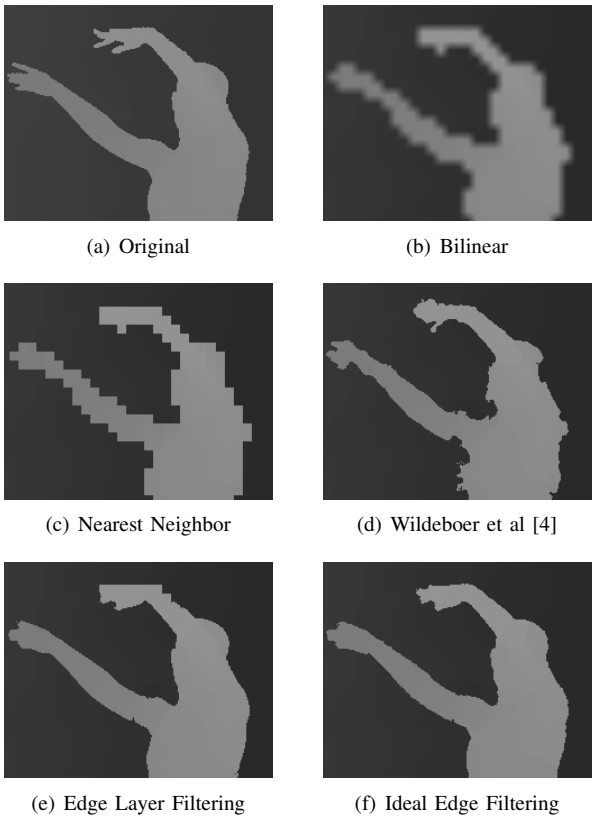
[1] A 3×3 square structuring element is used in our simulations

Since the unfiltered areas are usually smooth, the nearest neighbor interpolation is a satisfactory and a computationally-efficient up-sampling procedure.

For the pixels that belong to any of the identified edge contours, the following non-linear smoothing filtering is performed. Consider a 7×7 neighborhood of pixels around a center pixel to be filtered. The algorithm first identifies pixels within the neighborhood that belong to the same or nearby edge layer as the center pixel; only pixels that belong to the same type of edge layer, i.e., background or foreground layers, are considered. Then, a median filter is applied to the selected pixels. In this way, pixels assume smoother values similar to the ones along the edge direction. The median operation only uses depth values that are already present in the image, thereby eliminating outliers values.

The filtering procedure is done from the outer contours and works inward to the contours closer to the hypothesized depth discontinuity. The image is updated with the filtered pixels for each layer to provide a smoother neighborhood for pixels in layers that are closer to the edge origin. At the edge origin location, the pixel can belong either to the background or the foreground. In order to preserve the object boundary as much as possible, the edge location is assumed to belong to the foreground and is filtered with edge layers that includes the foreground.

Fig. 2 shows details of the filtered depth map of the Ballet sequence with subsampling factor of 8, using several different approaches. It is shown that the bilinear filter significantly blurs the edges, while nearest neighbor interpolation generates depth maps with jagged edges. Methods that use the texture edges, such as the one described in this paper or the one proposed by Wildeboer, et al. [4] are able to reconstruct edges with higher fidelity, but still suffer from artifacts caused by the mismatch between texture and depth maps. One way to overcome this is to apply the proposed method to edges extracted from the original depth image; we refer to these as ideal edges. As expected, improved results can be obtained using ideal edges as shown in Fig. 2(f). The performance tradeoffs in a rate-constrained framework are a subject of further study.



(a) Original (b) Bilinear

(c) Nearest Neighbor (d) Wildeboer et al [4]

(e) Edge Layer Filtering (f) Ideal Edge Filtering

Fig. 2. Depth Map of Ballet Sequence, view 5, frame 10

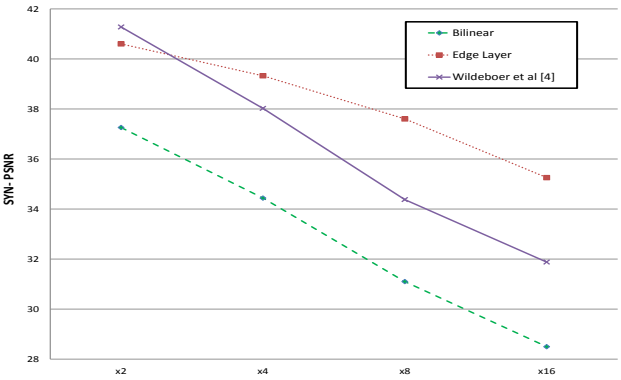


Fig. 3. Ballet Synthesis RD performance

V. Simulation Results

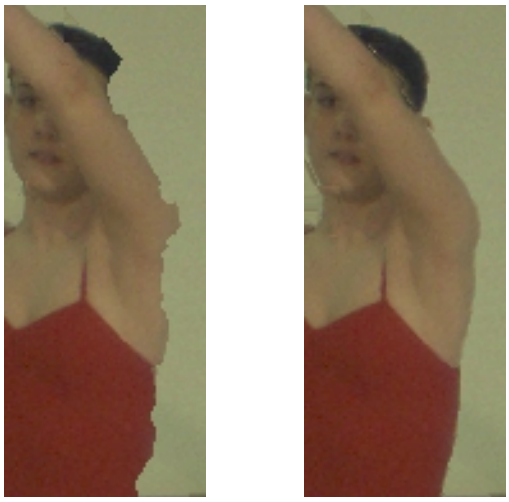
The performance of the proposed edge-layer up-sampling algorithm is evaluated on sequences with well-aligned depth maps (such as Ballet), as well as sequences with non-aligned estimated depth maps (such as Newspaper). The original depth images are down-sampled with a range of scaling factors and up-sampled. The up-sampled images are then used to generate synthesized views. Comparisons are made both objectively and subjectively on the synthesized images.

A. Performance with Varying Scale Factor

The first experiment assesses the performance at various scaling factors from 2 to 16. Fig. 3 shows the quality of the synthesized view from the Ballet sequence using different up-sampling methods. Compared to the bilinear filter, methods that use texture information have a much better performance. It is shown that the proposed technique is comparable to the texture-based up-sampling method of [4] for a scaling factor of 2, but outperforms this method for higher scaling factors. Fig. 4 shows the synthesized view obtained by both methods at a scale factor of 8, where it is evident that the proposed method yields higher subjective quality and better preserves the object boundaries.

B. Sensitivity to Edge Accuracy

The next experiment studies the sensitivity to the accuracy of the extracted edge information, which is important when



(a) Wildeboer et al [4] (b) Edge Layer

Fig. 4. Ballet synthesis, scaling factor 8.

the depth map is not aligned with the texture. The Newspaper sequence is used for this purpose. It is shown in Fig. 5 that the synthesis quality of the bilinear filter is higher than techniques that rely on texture information, including our proposed method and that of Wildeboer [4]. However, when using ideal edges with our proposed approach, a substantial increase in performance can be observed.

A subjective comparison of the results are shown in Fig. 6. It is clear from these image samples that the proposed technique achieves comparable quality to the bilinear filtering approach despite the 3dB difference in PSNR measured on the synthesized views. This discrepancy between objective measures and actual subjective quality is an ongoing area of research, and indicates that one cannot rely solely on objective metrics when evaluating the quality of synthesized views.

VI. Conclusions

A novel up-sampling technique for depth map images was presented that applied a non-linear filter to pixels considering edge layers extracted from the texture. It was shown that smoothing the depth values along the edge lines provides a better recovery of the object boundary and results in higher quality synthesis results compared to reference methods for a wide range of scaling factors. This paper also showed that improved results are possible with more accurate edge information, e.g., as extracted from the original depth map.

As part of our future work, we intend to further study the performance of the proposed method in the context of texture and depth map coding, including the performance tradeoffs when edge information is explicitly coded and transmitted, and the impact on coding efficiency when the depth information is utilized within the coding loop, e.g., for view synthesis prediction. Specifically, it is not yet known whether higher scaling factors for the depth coupled with improved up-sampling techniques provide benefits in terms of overall coding efficiency and rendering performance.

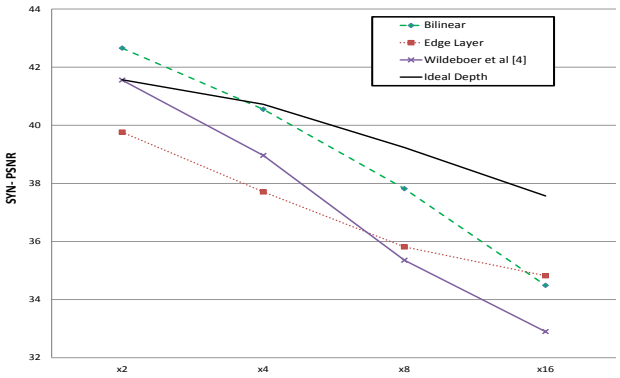
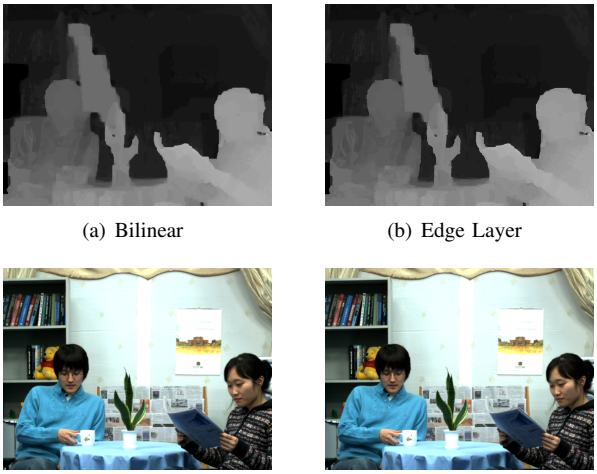


Fig. 5. Newspaper Synthesis RD performance



(a) Bilinear (b) Edge Layer

(c) Bilinear (42.65dB) (d) Edge Layer (39.76dB)

Fig. 6. Newspaper synthesis, scaling factor 2.

References

[1] A. Vetro, T. Wiegand, and G.J. Sullivan, "Overview of the Stereo and Multiview Video Coding Extensions of the H.264/AVC Standard," *Proc. of the IEEE*, vol. 99, no. 4, pp. 626 - 642, April 2011.

[2] K. Mueller, P. Merkle, and T. Wiegand, "3D Video Representation Using Depth Maps," *Proc. of the IEEE*, vol. 99, no. 4, pp. 643 - 656, April 2011.

[3] K.-J. Oh, S. Yea, A. Vetro, and Y.-S. Ho, "Depth reconstruction filter and down/up sampling for depth coding in 3D video," *IEEE Signal Processing Letters*, vol. 16, no. 9, pp. 747–750, Sept. 2009.

[4] M. Wildeboer, T. Yendo, M. Panahpour Tehrani, T. Fujii, and M. Tanimoto, "Depth up-sampling for depth coding using view information," *Proc. 3DTV-CON*, May 2011.

[5] E. Ekmekcioglu, M. Mrak, S. Worrall, and A. Kondoz, "Utilisation of edge adaptive upsampling in compression of depth map videos for enhanced free-viewpoint rendering," *Proc. ICIP*, pp. 733–736, 2009.

[6] S. Liu, P. Lai, D. Tian, and C. W. Chen, "New depth coding techniques with utilization of corresponding video," *IEEE Trans. Broadcasting*, vol. 57, no. 2, pp. 551–561, June 2011.

[7] J. Choi, D. Min, and K. Sohn, "2d-plus-depth based resolution and frame-rate up-conversion technique for depth video," *IEEE Trans. Consumer Electronics*, vol. 56, no. 4, pp. 2489–2497, Nov. 2010.

[8] A. Sanchez, G. Shen, and A. Ortega, "Edge-preserving depth-map coding using graph-based wavelets," in *Proc. 43rd Asilomar Conference on Signals, Systems and Computers*, pp. 578–582, Piscataway, NJ, 2009.

[9] G. Shen, W.-S. Kim, A. Ortega, J. Lee, and H. Wey, "Edge-aware intra prediction for depth-map coding," in *Proc. ICIP*, pp. 3393–3396, Hong Kong, China, 2010.