Lossy Depth Image Compression using Greedy Rate-Distortion Slope Optimization

Ionut Schiopu, *Student Member, IEEE*, and Ioan Tabus, *Senior Member, IEEE*

***Abstract*—We introduce a method to create lossy versions of one image, either by successively merging the constant regions of the original image, or by iteratively splitting the regions from a created lossy image using horizontal or vertical line segments. Merging and split decisions are greedily taken, according to the best slope towards next point in the rate-distortion curve. For each created lossy image, the region contours and the optimal depth values can be entropy coded in three ways: with a new algorithm, or with two existing lossless coding algorithms. The obtained results compare favorably with the existing lossy methods.**

***Index Terms*—Depth image, image coding, lossy depth compression, region merging, region splitting, sequence of lossy images.**

I. Introduction

The next generation technologies in the entertainment field are the 3DTv and the free-view point video (FFV), in which an important role has the compression of depth map images and sequences. Consequently, depth image compression is an active field, with many potential applications.

Lossy depth image compression was proposed in the past using several approaches. A first approach is to obtain a decomposition of the depth image using a binary tree triangular decomposition [1], or a quad-tree decomposition of the depth image with a wedgelet and platelet based approach that fills regions [2]. A second approach is to obtain a segmentation of the images: in [3] a segmentation of the color image is used for segmenting the depth image; in [4] the method starts from an over-segmented depth image and merges regions according to the average depth value for each region and the number of objects assumed to exist in the depth map; in [5] the depth image is segmented into objects, and the contour and the depth of the objects are compressed using various methods; in [6], [7] the image is compressed by two image pyramid structures, one for arc breakpoints and other for sub-band samples.

The letter presents an algorithm that generates sequences of lossy depth images for the full range of rates and proposes suitable entropy coders. The algorithm is not scalable, however suitable modifications can make it scalable by decreasing the performance. We show how to iteratively generate sequences of depth images using a greedy best slope criterion by merging regions, Section II-A, or splitting regions, Section II-B. Suitable entropy coding procedures are presented in Section II-C. Obtained results for compressing commonly used depth images are presented in Section III.

II. Description of the Greedy Slope Optimization (GSO) Method

In the GSO method we construct two sequences of lossy images, each image being composed of connected regions, and each region having the same reconstructed depth value.

The first sequence starts with the original image, partitioned into its constant connected components, and the next lossy images are generated by merging at each step the two regions, for which the slope of the rate-distortion (RD) curve is optimal. The greedy merging process results in a good trade-off rate-distortion for the medium and high rates, until the number of regions is in the order of tens. The most important image contours of the initial depth image are contained by the last lossy images in the sequence, any of them can be selected to define the template for the second phase. This first phase sequence ends with a lossy image with only two regions.

The second sequence of lossy images is obtained starting from the chosen template, and advances by splitting the regions, using horizontal or vertical line segments. For each obtained lossy image we encode the contours of the regions and the depth value of each region using entropy coding. In general the contours of the regions are encoded using chain-codes, except the straight line segments which are encoded more efficiently according to their position inside a region.

Each depth image, Z, is a matrix with n_r rows and n_c columns. An integer $Z(x,y) \in \{0,1,\ldots,2^B-1\}$ is stored for each pixel (x,y), using B bits, representing the distance between the camera lens and a point in the scene. In this letter we illustrate the method over images containing integers stored using $B=8$ b.

A. GSOm: GSO With Region Merging

We denote Z the current lossy reconstruction of the original depth image Z_0, and explain how the next image in the sequence, Z', is constructed.

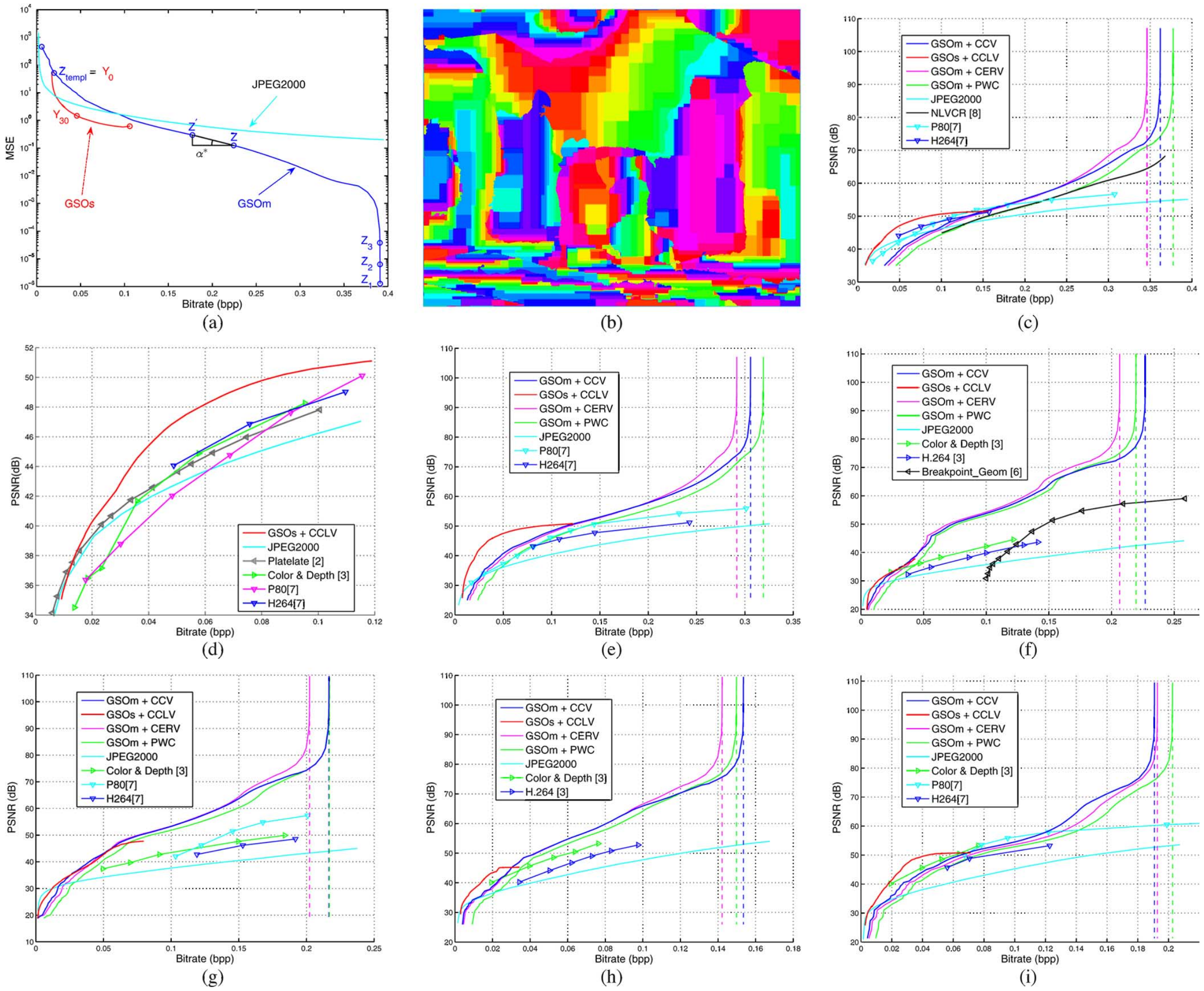
The image Z is partitioned as $\cup_{i=1}^{n_\Omega}\Omega_i$, where each of the n_Ω regions is a set of pixels connected in 4-connectivity. Initially, the partition of Z_0 includes all maximal constant regions in the image. A maximal region Ω_i has the same depth value d_i for all its pixels, and every pair of neighboring regions has distinct depth values. The partition into regions is efficiently encoded using the 3OT chain-code representation of the crack-edges separating the neighboring pixels belonging to different regions [8]. For each pair of neighboring regions (Ω_i,Ω_j) their common contour segment is denoted $\Gamma_{i,j}$, formed of 3OT chain-

Manuscript received June 20, 2013; revised August 16, 2013; accepted August 24, 2013. Date of publication September 04, 2013; date of current version September 11, 2013. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Ce Zhu.

The authors are with the Department of Signal Processing, Tampere University of Technology, Tampere, Finland (e-mail: ionut.schiopu@tut.fi; ioan.tabus@tut.fi;url: http://www.cs.tut.fi/~schiopu/; http://www.cs.tut.fi/~tabus/).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/LSP.2013.2280495



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Fig. 1. (a) The rate-distortion points corresponding to the sequence of lossy images for *Breakdancers*: GSOm, with blue; GSOs, with red; JPEG2000 with cyan. From the generic image Z to the next image Z' the merging pair is chosen so that the angle α is minimized, obtaining α^* and corresponding slope λ^*. (b) Segmentation for a lossy image from GSOs, using Z_{templ} with nine regions, for the initial image *Breakdancers*, at Bitrate $= 0.039$ bpp, PSNR $= 45.215$ dB. (c)–(i) PSNR vs Bitrate plots for the two sequences, GSOm and GOSs, compressed using the entropy coders: CCV, CCLV, CERV, and PWC, and results previously reported in literature, for a set of 6 images. For each plot, the listed BD-PSNR [13] value is computed to show the average improvement in PSNR of GSOm + CCV (GSOs + CCLV) with respect to the best previously reported method, which is named Reference Method. The results are (c) *Breakdancers*, BD-PSNR= 2.89 dB, ref. met. [7], (d) *Breakdancers*, zoom for [0, 0.12] bpp, BD-PSNR= 1.72 dB, ref. met. [2], (e) *Ballet*, BD-PSNR= 7.08 dB, ref. met. [7], (f) *Art*, BD-PSNR= 6.49 dB, ref. met. [3], (g) *Aloe*, BD-PSNR= 10.63 dB, ref. met. [3], (h) *Baby*, BD-PSNR= 4.64 dB, ref. met. [3], (i) *Bowling*, BD-PSNR= 3.48 dB, ref. met. [3].

codes. The partition is encoded using the sequence of chain-codes $\Gamma(Z)$, which is the union of all $\Gamma_{i,j}$.

The next image in the sequence, Z', is obtained from Z, by merging the pair $(\Omega_{i^*}, \Omega_{j^*})$. The new partition is specified by the contours $\Gamma(Z')$, having one segment less, $\Gamma(Z') = \Gamma(Z) \setminus \Gamma_{i^*,j^*}$. For brevity the pairs of region indices are denoted $p = (i, j)$. When merging two regions we aim at getting the best trade-off between the estimated saving of bits ΔR, due to not encoding Γ_p, and the increase in distortion ΔD, due to having a poorer reconstruction in the common region $\Omega'_\ell = \Omega_i \cup \Omega_j$. The unique depth value d'_ℓ is optimally set as rounded mean value of the depth inside the region Ω'_ℓ. In the rest of the letter the reconstructed depth for Ω_i or Ω'_i is d_i or d'_i respectively.

In order to select the optimal merging we evaluate ΔR by using a model for the contour codelength, $C(\Gamma_p) = C_1 \cdot L(\Gamma)$, where C_1 is a constant representing the cost of encoding a chain-code link and $L(\Gamma_p)$ is the number of links in the contour Γ_p. In this letter we used $C_1 = 1.5$ b and for the cost of encoding a depth value we have used the estimate $C_2 = 8$ b. The actual entropy coding will produce slightly different values, but they are not known at this stage. We estimate the rate variation, ΔR, when we eliminate the contour Γ_p:

$$\Delta R_p = C_1 \cdot L(\Gamma_p) + C_2. \qquad (1)$$

The change in distortion $\Delta D = MSE(Z') - MSE(Z)$ depends only on the regions involved in merging: $\Omega_i \cup \Omega_j \to \Omega'_\ell$, due to the iterative nature of constructing the images Z and Z'. In order to evaluate efficiently ΔD, we introduce for any region Ω_k with m_k pixels the following variables: the sum of original depth values $\phi_k = \sum_{(x,y)\in\Omega_k} Z_0(x,y)$, the sum of squared depth, $\varphi_k = \sum_{(x,y)\in\Omega_k} Z_0(x,y)^2$, and the sum of squared reconstruction errors $\rho(\Omega_k) = \sum_{(x,y)\in\Omega_k} (Z_0(x,y) - d_k)^2 = \varphi_k - 2\phi_k d_k + m_k d_k^2$, resulting in the mean square error over the region $MSE_k = \frac{1}{n_r n_c}\rho(\Omega_k)$ and $MSE(Z) = \sum_{k=1}^{n_\Omega} MSE_k$. The merging of the pair of regions (Ω_i, Ω_j) has the following consequences: the contour Γ_p is removed; a new region Ω'_ℓ is created, having the variables $m'_\ell = m_i + m_j$, $\phi'_\ell = \phi_i + \phi_j$, $d'_\ell = \lfloor \phi'_\ell / m'_\ell \rceil$; a change in distortion is introduced:

$$\Delta D_p = \frac{\rho\left(\Omega'_\ell\right)}{n_r n_c} - \frac{\rho(\Omega_i) + \rho(\Omega_j)}{n_r n_c} = \frac{m'_\ell d'^2_\ell - m_i d_i^2 - m_j d_j^2 + 2\left(d_i \phi_i + d_j \phi_j - d'_\ell \phi_{\ell'}\right)}{n_r n_c}. \quad (2)$$

The pair of regions to merge, $(\Omega_{i^*}, \Omega_{j^*})$, is chosen from all available pairs of neighbor regions, by greedily minimizing the slope between the points Z and Z' in the RD plot (Fig. 1(a)), i.e. by choosing the pair $p^* = (i^*, j^*)$, for which

$$\lambda_p = \tan \alpha_p = \frac{\Delta D_p}{\Delta R_p} \quad (3)$$

is minimum, resulting in the smallest slope $\lambda^* = \tan \alpha^*$.

Although the number of available pairs of neighboring regions n_Γ in the image Z_k (the same as the number of contour segments at the current step) is large for the first images, when moving from Z_k to Z_{k+1} the value of λ_p will not change, except for the pairs formed by the new region Ω'_ℓ with its neighbors. To avoid making updates after each merging step, the values of λ_p for all possible region pairs are sorted increasingly in the vector $\boldsymbol{\lambda} = [\lambda_{p_1}, \lambda_{p_2}, \ldots, \lambda_{p_{n_\Gamma}}]$, corresponding to the list of pairs of regions truncated to the first n_M pairs, $[(i_1, j_1), (i_2, j_2), \ldots, (i_{n_M}, j_{n_M})]$. The pairs from the list are marked for merging, starting from the first and continuing sequentially, except for pairs (i_k, j_k) having either i_k or j_k an element in an earlier pair $(i_{k-\tau}, j_{k-\tau})$. Additionally, when the size of a region Ω_{i_k} is smaller than three pixels, the index j_k is allowed to appear in following marked pairs. In this letter we used $n_M = 100$, if $n_\Omega / \sigma > 100$; $n_M = \lfloor n_\Omega / \sigma \rfloor$, if $1 \le n_\Omega / \sigma \le 100$; and $n_M = 1$, if $n_\Omega / \sigma < 1$; where $\sigma = 10$.

For obtaining operating points in the RD curve near the lossless rate, we apply a different method, not by merging regions, but by modifying slightly the contours as follows: if at least three neighbors of a current pixel with depth d_i, have the same depth value, d_j, and if $|d_i - d_j| \le 2$, then we set the depth value of the current pixel as d_j. The process is done columnwise sequentially and intermediate images are stored along the process when MSE reached an empirically selected value, so that $\Delta PSNR \in [3, 7]$.

B. GSOs: GSO With Region Splitting

For the second phase (GSOs), we start from one of the images obtained in GSOm, where the number of regions is small (here we used templates with two and nine regions) and call this image a template, $Z_{templ} = Y_0$ (see Fig. 1(a)). From each such template, which corresponds to very low bitrates in the RD curve obtained for GSOm, we create a sequence of images, $Y_0, Y_1, \ldots$, where the regions of the image Y_k are further split with horizontal or vertical lines to obtain the regions of the image Y_{k+1}. To recreate the contour at the decoder we first encode the contours of Y_0, the same way as for GSOm, and additionally we encode: the decisions to split or not a region; the orientation (vertical or horizontal); and location of the line segment for each decided split.

The image Y_{k+1} is obtained from the image Y_k, by fixing a target slope, λ_{k+1}, and splitting all the regions iteratively, starting from the regions of Y_k until no more splits are allowed at given slope λ_{k+1}, as explained below. The decisions to split form a binary tree, traversed in depth-first order. The greedy decisions are choosing between splitting with a vertical line segment at a position J or a horizontal line segment at position I from all possible positions determined by the boundaries of the region Ω_i ($I_{min} \le I < I_{max}$, $J_{min} \le J < J_{max}$). The resulting regions are denoted $\Omega^I_{i_1}$ and $\Omega^I_{i_2}$, for horizontal splits, while $\Omega^J_{i_1}$ and $\Omega^J_{i_2}$, for vertical splits.

The criterion to be maximized is the slope, in the RD plot, from the set of slopes for all possible splits:

$$\Psi = \left\{ \frac{\Delta D_h(I)}{\Delta R_h(I)} \right\}_{I_{min} \le I < I_{max}} \bigcup \left\{ \frac{\Delta D_v(J)}{\Delta R_v(J)} \right\}_{J_{min} \le J < J_{max}},$$

where the changes in distortion are evaluated as $\Delta D_h(I) = \rho(\Omega_i) - \rho(\Omega^I_{i_1}) - \rho(\Omega^I_{i_2})$, $\Delta D_v(J) = \rho(\Omega_i) - \rho(\Omega^J_{i_1}) - \rho(\Omega^J_{i_2})$, and the additional rate is estimated as $\Delta R_h(I) = C_3 + \log_2(I_{max} - I_{min})$, $\Delta R_v(J) = C_3 + \log_2(J_{max} - J_{min})$ for horizontal, respectively vertical split. The constant C_3 includes the cost of an additional depth value needed after the split and the cost for encoding the decision. In this letter we use $C_3 = 8$ b, although the true encoding process uses adaptive Markov modeling and obtains better rates, but the choice of C_3 was found to not influence the results too much. If the maximum slope λ^* from the set Ψ is higher than the target slope, $\lambda^* > \lambda_{k+1}$, we split the region with a horizontal or vertical line segment, by encoding the decision and the row or column index selected according to the maximum slope λ^*. The decisions at each region are represented with the variable ξ as follows: $\xi = 1$ for no-split decision ($\lambda^* \le \lambda_{k+1}$); $\xi = 2$ for vertical split; and $\xi = 3$ for horizontal split. The sequence of variables ξ collected along the split process at the encoder is transmitted to the decoder who can reproduce the splitting process. The variables ξ are encoded using order two adaptive Markov arithmetic coding.

If $\xi = 1$, we compress d_ℓ using the up and left neighboring region values (see Section II-C). If $\xi = 2$, we compress the row index I^* using $\log_2(I_{max} - I_{min})$ bits, split the current region into two regions having same column indices and the following row indices: $[I_{min}, I^*]$ for $\Omega^I_{i_1}$; $[I^* + 1, I_{max}]$ for $\Omega^I_{i_2}$. The case $\xi = 3$ for compressing and performing the vertical splits is similar to the previous case. The process continues by applying the algorithm first for Ω_{i_1} and all its descendant regions, until no more splits occur in this branch, and only then Ω_{i_2} is processed similarly.

For minimizing the distortion, the reconstruction value in each region is assigned as follows: the depth value $d_\ell - 1$ is set for the first line and the first column of the current region if the neighboring region value d_i is $d_i < d_\ell$; similarly it is set to $d_\ell + 1$ if $d_i > d_\ell$. For a better compression the true contour is smoothed as in Section II-A, this time using the constraint $|d_i - d_j| < \gamma$, where γ increases until the value 100.

In Fig. 1(a) we present a RD plot that illustrates the principle of the method, using real points corresponding to the generated sequences of lossy images for *Breakdancers* (including also JPEG2000 results), and the angle α^* that minimized the slope λ^* for generating the next image Z'. Fig. 1(b) shows an example of segmentation for a lossy image from GSOs, using the template with nine regions, for the initial image *Breakdancers*, compressed at 0.039 bpp and having PSNR $= 45.215$ dB. In Fig. 1(b) can be seen the two types of contours: straight lines (vertical and horizontal) and non-straight true object boundaries. In our experiments we combine the RD curves S_1 (obtained when starting from the template having nine regions) and S_2 (having a starting template with two regions). The transition from S_1, which is better in the range of rates [0.025, 0.1] bpp, to S_2 is performed empirically around the point having the slope $\lambda = 200$.

C. Entropy Coding

For compressing the contour for the sequence of images $Z_1, Z_2, \ldots$, we used a similar algorithm as in [8], to which we added improvements regarding the searching of the next position of the contour. The encoder transmits a matrix containing anchor points and junction points for 3OT chain-codes, and the all 3OT chain-codes. Here we modified the order in which the 3OT segments are concatenated, obtaining a better compression by using the following approach: start generating a new 3OT chain code segment by choosing the next link with the priority list: right, up, left, down. We refer to [8] for the detailed algorithm of encoding the region contours.

For encoding the depth values from the sequence of images $Z_1, Z_2, \ldots$, we use the method presented in [9], where the depth value of a current region, d_i, is encoded using its position in the list of possible depth values, ζ, generated using the known values $\mathbf{d}$ of the neighboring regions. The contour of Z_k is obtained using 3OT chain-codes which guarantees that $d_i \notin \mathbf{d}$. The initial algorithm excludes $\mathbf{d}$ from ζ, therefore d_i is encoded using $\zeta' = \zeta \cap \mathbf{d}$. This entropy coding method for contours and depth values is dubbed Chain-Code-Value (CCV).

For encoding the depth values from the sequence of images $Y_0, Y_1, \ldots$, a modified version of the algorithm is used: d_i is compressed using the up and left neighboring region values, and because the horizontal and vertical lines do not guarantee $d_i \notin \mathbf{d}$, d_i is encoded using ζ (not ζ'). We denoted this modified algorithm Chain-Code-Line-Value (CCLV).

III. Experimental Results

In this letter we present comparative results for six commonly used depth images: *Breakdancers*, frame 0 of cam0 from *breakdancers* dataset [10]; *Ballet*, frame 96 of cam0 from *ballet* dataset [10]; *Art, Aloe, Baby1* and *Bowling1*, full-size resolution, left view (disp1.png) from *Middlebury* dataset [11].

The algorithms were implemented in C. For the arithmetic coding routines we used the implementation from Witten *et al.* For GSOm a new lossy image is saved when $\Delta PSNR > 0.5$ dB between two consecutive lossy images or if $n_\Omega < 10$; while for GSOs, 50 values (selected empirically) are used for the stop criterion λ_k so that $\Delta PSNR > 0.1$ dB. Regarding the runtime, the current not optimized version of the method generates the last image for GSOm in 1.2 s for *Breakdancers* (1.9 s for *Aloe*) and compresses a Y_k image in less then 0.65 s.

In Fig. 1(c)–(i)[1] the GSOm sequence is compressed using three entropy coders: CCV; a recent algorithm for lossless compression of depth image, Crack-Edge-Region-Value (CERV) [9]; a palette image coder, the piecewise-constant image model (PWC) [12]; the GSOs sequence is compressed using CCLV. The results are compared also with: the Platelet algorithm [2]; an algorithm which uses color information [3], denoted here "Color & Depth"; the Breakpoint_Geom algorithm [6]; the "Proposed_80" algorithm from [7], denoted here "P80"; H.264 from [6] and [7]; our previous results [8]; the JPEG2000 standard. The Fig. 1(d) shows the results for the *Breakdancers* image for low bitrates, below 0.12 bpp. The figure shows that $\mathrm{GSOs} + \mathrm{CCLV}$ and entropy coding the GSOm sequence obtain the best results comparing with other algorithms listed above. The CCV algorithm is not the best lossless compressor but under 65 dB obtains the best, or similar, results comparing with the CERV algorithm.

IV. Conclusions

The GSO algorithm presented in this letter uses the greedy slope optimization for generating sequences of lossy images by merging (GOSm) or splitting (GOSs) regions. The results showed that the compression of the sequences using suitable entropy coder (CCV/CCLV, CERV, PWC) obtains better results over the full range of bitrates, when compared with previously reported results in literature.

References

[1] G. Carmo, M. Naccari, and F. Pereira, "Binary tree decomposition depth coding for 3D video applications," in *Proc. IEEE ICME*, Barcelona, Spain, Jul. 2011.

[2] Y. Morvan, D. Farin, and P. H. N. de With, "Depth-image compression based on an R-D optimized quadtree decomposition for the transmission of multiview images," in *Proc. IEEE ICIP*, San Antonio, TX, USA, Sep. 2007, vol. 5, pp. V-105–V-108.

[3] S. Milani, P. Zanuttigh, M. Zamarin, and S. Forchhammer, "Efficient depth map compression exploiting segmented color data," in *Proc. IEEE ICME*, Barcelona, Spain, Jul. 2011.

[4] S. Milani and G. Calvagno, "A depth image coder based on progressive silhouettes," *IEEE Signal Process. Lett.*, vol. 17, no. 8, pp. 711–714, Aug. 2010.

[5] B. Zhu, G. Jiang, Y. Zhang, Z. Peng, and M. Yu, "View synthesis oriented depth map coding algorithm," in *Proc. APCIP*, Jul. 2009, vol. 2, pp. 104–107.

[6] R. Mathew, P. Zanuttigh, and D. Taubman, "Highly scalable coding of depth maps with arc breakpoints," in *Proc. Data Compress. Conf*, Apr. 2012, pp. 42–51.

[7] R. Mathew, D. Taubman, and P. Zanuttigh, "Scalable coding of depth maps with r-d optimized embedding," *IEEE Trans. Image Process.*, vol. 22, no. 5, pp. 1982–1995, May 2013.

[8] I. Schiopu and I. Tabus, "Lossy and near-lossless compression of depth images using segmentation into constrained regions," in *Proc. 20th EUSIPCO*, Bucharest, Aug. 2012, pp. 1099–1103.

[9] I. Tabus, I. Schiopu, and J. Astola, "Context coding of depth map images under the piecewise-constant image model representation," *IEEE Trans. Image Process.*, to be published.

[10] C. Zitnick, S. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," in *Proc. ACM SIGGRAPH*, 2004, pp. 600–608, LA, CA.

[11] H. Hirschmuller and D. Scharstein, "Evaluation of cost functions for stereo matching," in *Proc. IEEE Comput. Soc. Conf. CVPR*, Minneapolis, MN, USA, Jun. 2007, pp. 1–7.

[12] P. Ausbeck, Jr., "The piecewise-constant image model," *Proc. IEEE*, vol. 17, no. 8, pp. 1779–1789, Nov. 2000.

[13] K. Senzaki, BD-PSNR/Rate Computation Tool for Five Data Points, JCT-VC of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Jul. 2011.

[1] For more results and some lossy images see www.cs.tut.fi/~schiopu/GSO