3D VIDEO CODING VIA MOTION COMPENSATION OF SUPERPIXELS

S. Milani and G. Calvagno

Dept. of Information Engineering, University of Padova,
e-mail: {simone.milani,calvagno}@dei.unipd.it.

ABSTRACT

Most of the past and current video coders partition the input frames into regular blocks of pixels that are approximated by a motion estimation unit and coded via a block-based transform. A better performance can be obtained by adapting the size of the approximated region to the geometry and the characteristics of the objects captured by the camera.

The paper presents a novel coding scheme for video+depth signals that combines a 3D object identification unit with an object-oriented motion estimation strategy. Object identification is obtained via a joint luminance-depth oversegmentation of the acquired scene which partitions the input scene into superpixels. The procedure can be easily replicated at the decoder, and therefore, does not imply the coding and transmission of object masks. The scheme outperforms the rate-distortion performance of H.264/AVC of 2 dB with a reasonable increment of the complexity for motion estimation and segmentation.

1. INTRODUCTION

During the last years, the widespreading of 3D video displays and applications has finally opened the way to the long-waited 3D video revolution. Movies, television broadcasting, gaming and video communication applications are adopting more and more frequently 3D video technology in order to increase the level of immersiveness and involvement of users. As a consequence of this diffusion, effective 3D video compression schemes are required in order to face two of the most troublesome issues about 3D video delivery. One is the need to transmit a significantly-bigger amount of data with respect to traditional video communication. The other is the need to process different signals with heterogeneous characteristics, e.g., 3D video includes standard video signals and geometry data. A widely-adopted format for 3D video sequences is the Depth Based Image Rendering or DIBR [1], which associates a depth map to each frame of a standard video sequence.[1].

Despite the fact that depth signals present peculiar characteristics that prove to be completely different from standard video and could be better exploited by ad-hoc coding solutions [2, 3], depth and color (or texture) sequences show a strong correlation in the motion flow that can be exploited in joint source coding [4, 5] or bit allocation [6].

Moreover, a joint processing of video and depth signals permits identifying more accurately the objects that are present in the scene. This capability provides a significant leap forward in achieving high compression gains since a small amount of data can characterize a wide region in the captured scene (corresponding to a single object) whenever the motion or spatial correlation present the same characteristics all over it. In the latest video coding standards, the possibility of characterizing the movement for wide areas is implemented by enabling motion estimation (ME) and compensation on variable-sized blocks that can partially match the elements in the scene since adaptation permits mitigating the compression inefficiency derived from the unnatural partition of the image to be coded into square or rectangular blocks. An example for this can be found in the coding standard H.264/AVC [7] which supports several block partitions from 4×4 to 16×16 pixels. More recently, the current standard HEVC [8] allows prediction blocks sized up to 64×64 pixels. The choice of the most proper block size is performed according to its rate-distortion performance on the pixel region to be coded.

A more flexible partitioning can be obtained via segmentation, which improves significantly the compression gain. In fact, grouping set of arbitrarily-shaped pixel regions enables their approximation via spatial or temporal prediction characterized by a common parameter set (e.g., the same prediction orientation or motion vectors). In addition, recent results on the compression of geometry data (such as depth maps, meshes) has shown that coding schemes based on segmentation prove to be more effective than traditional image coding strategies [2]. One reason for such improvement has to be found in the peculiar nature of geometry signals (e.g., depth maps) which present an alternation of smooth areas and sharp edges. As a matter of fact, using segments that fit the edges permits improving the compression efficiency. Initial works on segmentation-based video coding were started more than 15 years ago (see [9] and [10]). The possibility of characterizing individual objects in a coded scene was later standardized within the following video coding standard like MPEG-4 [11], but it was never widely adopted and, so far, few coding systems take full advantage of object-oriented coding architectures [12]. This fact is due partly to the computational cost of segmentation, partly to the need of signaling object shapes to the decoder (which implies the coding and transmission of binary or alpha masks), and partly to the need of precision in the segmentation process (which conditions an accurate object identification).

However, the recent widespreading of 3D video and the possibility of obtaining depth maps in real time, e.g., via Time-of-Flight (ToF) sensors or structured light cameras (see [13]), has provided video coding designer with novel signals that can be processed to obtain an accurate partitioning of the acquired scene at a reasonable computational cost. In fact, joint segmentation of texture and geometry signals permits identifying the borders of the objects accurately so that edge distortion is limited and regions do not overlap over two different objects. This latter inconvenient is utterly minimized by forcing the segmentation routine to generate a significant number of small segments (oversegmentation), called *superpixels* [14].

This paper presents a video coding system that replaces the traditional block-based motion compensation with a superpixel-based motion compensation, where superpixels are obtained via a joint segmentation strategy processing both texture and depth signals. In this part, different algorithms (such as k-means [15] or the one by Felzenszwalb and Huttenlocher [16]) were tested and adapted to handle DIBR data. Superpixels[2] are, at first, generated on the previously-coded frames and, then, propagated to the following frame exploiting the correlation existing among adjacent pictures. For each segment in the current DIBR image (including both texture and depth components), a motion estimation unit computes a predictor segment in the previous frame that minimize a cost function, and characterizes it via a motion vector (MV). The proposed approach permits reducing the overall bit rate significantly since the segmentation can be reconstructed faithfully at the decoder from a minimal set of transmitted data, and it is possible to

[1] The DIBR format can be referenced as video+depth format as well.

[2] In the paper, the terms "superpixels", "segments", and "partitions" are used equivalently to denote the image areas identified by the segmentation strategy.

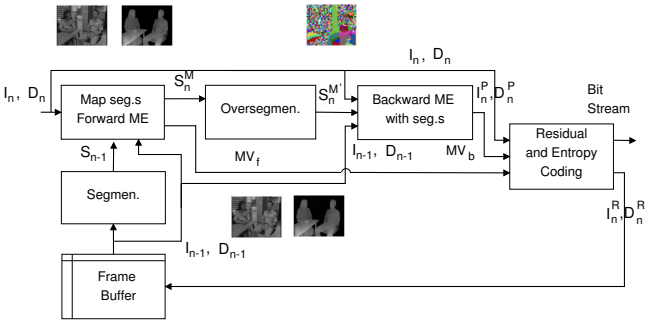

Figure 1: Encoder block diagram.

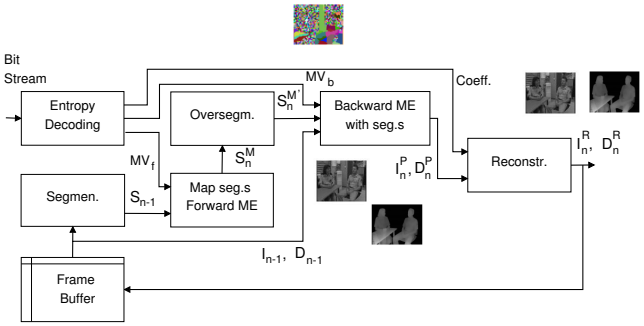

Figure 2: Decoder block diagram.

employ arbitrarily-shaped blocks in motion compensation (reducing the amount of coded MVs and approximating the real motion of objects more accurately). Experimental results show that the proposed strategy obtains a significantly-better rate-distortion performance with respect to the standard H.264/AVC strategy.

In the following, Section 2 presents the structure of the coder, with Subsection 2.1 reporting the adopted segmentation strategy, Subsection 2.2 showing how segments are used to predict the current frame, and Subsection 2.3 describing the entropy coding strategy. Experimental results are reported in Section 3, and final conclusions are drawn in Section 4.

2. STRUCTURE OF THE CODING/DECODING ARCHITECTURE

The proposed video coder mainly differs from previous hybrid video coding standards in the motion estimation and compensation routine, where a segmentation-based temporal prediction of the current frame is performed. In the implemented scheme, two coding types are supported for each coded frame: Intra coding and Inter coding. Intra coding correspond to the Intra coding strategy defined within the standard H.264/AVC. As for Inter frames, the coding process can be split into three steps or phases. During the first phase, the codec computes a segmentation mask for the current frame such that objects in the scene can be identified by disjoint sets of small segments (superpixels). In a second step, the ME unit approximates each superpixel with an identically-shaped predicting region found in the previously-decoded frame. Finally, the prediction residual signal is processed by a traditional transform coding scheme. Both MVs and transform coefficients are then coded into a binary stream and transmitted to the decoder.

The whole coding scheme for Inter frames is depicted in Figure 1, while in Fig. 2 the block diagram of the corresponding decoder is shown.

At the beginning of each group of pictures (GOP), the first video and depth frames are coded using the Intra strategy defined in H.264/AVC. The reconstructed pictures are then buffered in the frame buffer of Fig. 1, and the Inter coding process is adopted for the following frames until the end of the GOP.

In the following, the different steps of the Inter coding strategy will be described in detail.

2.1 Segmentation strategy

The first step of the Inter coding process consists in partitioning the input frames into superpixels that can be used by the ME unit to effectively compute a prediction for the current frame. The size of superpixels proves to have a crucial role in the rate-distortion performance of the coder since the effectiveness of ME depends on the accuracy in matching the segments to the object in the scene. Using small superpixels reduces the probability that they overlap over two different objects and increases the performance of temporal prediction. On the other hand, having many small superpixels in the scene increases the bit rate required to code the MV field since a motion vector needs to be signalled for each segment. As a matter of fact, the coder needs to find an appropriate trade-off matching the characteristics of the coded sequence.

Moreover, the segmentation needs to be signalled to the decoder without implying an undesirable increase in the coded bit rates. The additional bit rate that is required for the coding of binary or alpha object masks [11] can not be overlooked in the overall bit budget for the whole sequence, and therefore, a processing strategy that permits deriving the segmentation from the available data with limited additional information has to be preferred. As a consequence, this choice implies an increase in the complexity of the decoder since the derivation of the image segmentation has to be performed in reconstructing the sequence as well.

The approach adopted in the proposed coder can be divided into three steps. An initial segmentation is performed on the previously-reconstructed texture and depth frames obtaining the segmentation layout S_{n-1} (see Fig. 1, 2, and 5). Then, the segmentation S_{n-1} is propagated to the following frame, and finally, some of the superpixels are utterly split into smaller superpixels in case a finer partitioning is required. In the following these three phases are explained in detail.

2.1.1 Joint video+depth segmentation of the previously-decoded frames

Let I_n and D_n be the input texture and depth frames at the time instant n, and let $\hat{I}_{n-1}$ and $\hat{D}_{n-1}$ the reconstructed frames at time instant $n-1$ that are stored in the frame buffer. The previously-coded frames $\hat{I}_{n-1}$ and $\hat{D}_{n-1}$ are used as "side information" and partitioned into superpixels $R_{n-1,k}$, $k = 0,\ldots,N_S-1$, using a joint video+depth segmentation algorithm. In the paper, the superpixels $R_{n-1,k}$ are represented by sets of coordinates (x,y) related to the included pixels ($R_{n-1,k} \in \mathbf{Z}^2$). The number N_S of initial superpixels depends on the image resolution and on the adopted segmentation algorithm.

In our approach, we tested different segmentation algorithms in order to find the best solution in terms of accuracy and required complexity. At first, we partitioned the input frames into disjoint superpixels following a k-means clustering algorithm [15] where the classified array $\mathbf{a}(x,y)$ associated to the pixel position (x,y) is

$$\mathbf{a}(x,y) = [\,\hat{I}_{n-1}(x,y)\;\hat{D}_{n-1}(x,y)\;x\;y\,] \qquad (1)$$

where $\hat{I}_{n-1}(x,y)$ and $\hat{D}_{n-1}(x,y)$ are respectively the depth and the luminance values at position (x,y).[3]

In a second step, we considered the solution presented in [16], which is based on a graph-cut strategy and was appropriately modified in order to support DIBR signals. More precisely, the weights

[3]From this point on, we will use the symbols I_n as reference to both the image and the luminance component of the texture signal at time instant n. This simplification is here allowed since the current implementation of the coder supports gray scale sequences.

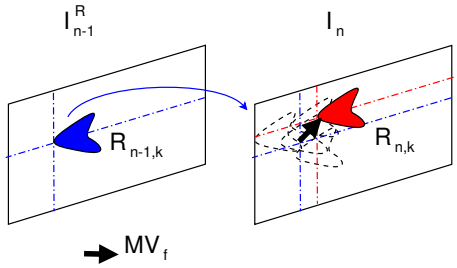

Figure 3: Estimation of MV_f.

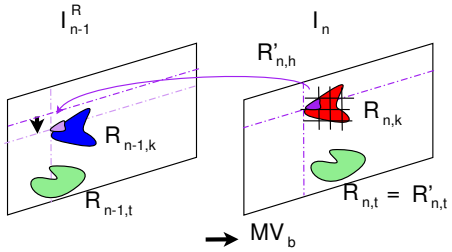

Figure 4: Partitioning of $R_{n,k}$ and estimation of MV_b.

of the edges in the graph, which are computed averaging the absolute differences between the RGB components of adjacent pixels after a low-pass filtering, are modified including in the average the absolute difference between depth values as well (where no smoothing is performed on the input depth signal).

In the following, we will describe how the algorithm propagates the partitioning S_{n-1} on the following frames I_n and D_n.

2.1.2 *Propagation of segmentation*

The temporal correlation existing between adjacent frames suggests that the same regions $R_{n-1,k}$ could be found in the frames I_n and D_n arranged at different positions because of motion. As a matter of fact, a forward motion estimation routine associates each superpixel $R_{n-1,k}$ in $\hat{I}_{n-1}$ ($\hat{D}_{n-1}$) with a region $R_{n,k}$ in I_n (D_n) via a region matching algorithm minimizing a cost function. The superpixel $R_{n,k}$ presents the same shape of $R_{n-1,k}$ and, given $R_{n-1,k}$, it can be identified by a motion vector $\mathbf{v} = [v_x\ v_y]$, i.e., $R_{n,k} = \mathbf{v} + R_{n-1,k}$. The cost function is

$$C\left(R_{n-1,k}, R_{n,k}\right) = \sum_{(x,y)\in R_{n-1,k}} \left|\hat{I}_{n-1}(x,y) - I_n(x+v_x, y+v_y)\right| + \sum_{(x,y)\in R_{n-1,k}} \left|\hat{D}_{n-1}(x,y) - D_n(x+v_x, y+v_y)\right|, \quad (2)$$

which combines two Sum-of-Absolute-Differences (SAD) for both the luminance and the depth components of the corresponding regions. All the $R_{n,k}$ can be signalled by specifying $\mathbf{v}$ for each region (in Fig. 1 and 2 the new segmentation is referenced as S_n^M). In the paper (see Fig. 1 and 2) we would refer to these motion vector set with MV_f (see the graphic example of Fig. 3). In the paper each superpixel $R_{n,k}$ will also be called "superpixel" and will be the area on which motion compensation is performed.

Experimental results show that this prediction does not permit an effective motion compensation since some pixels of I_n could not be included in any of the regions $R_{n,k}$ (outliers). As a matter of fact, it is possible to overcome this problem with two filling strategies. In case a small segment of outlier pixels lies between two spatially-adjacent regions $R_{n,k}$ and $R_{n,k+1}$, $R_{n,k}$ and $R_{n,k+1}$ are enlarged in order to include them. A similar strategy is adopted whenever the ouliers fall between image borders and region $R_{n,k}$. Whenever the outlying area is too wide, i.e., the number of pixels is larger than a threhold, additional 4×4 square regions $R_{n,k}$, $k > N_S$, are added in the segment map to encapsulate the outlying pixels. This procedure is also adopted to refine the segmentation to obtain a better ME. More details will be provided in the next subsection. The final map will be made of arbitrarily-shaped superpixels derived from the image segmentation and square superpixels added in order to include all the pixels. Note that neither enlarging $R_{n,k}$ nor adding extra regions requires signalling additional information in the bit stream since the decoder is able to reproduce the final partitioning from the sole set MV_f.

2.1.3 *Refinement of the segmentation*

The use of $R_{n,k}$ prevents an effective motion compensation since segmentation is performed on a reconstructed image after compression, and therefore, the identification of segments is affected by coding artifacts and distortion. As a consequence, the segmented area may result too wide and could include different objects. It is possible to improve the motion compensation performance by dividing the superpixels $R_{n,k}$ into smaller segments. Whenever the measured SAD proves to be higher than a threshold value, the pixels $\mathbf{p} = (x,y)$ in the segment $R_{n,k}$ are divided into smaller blocks $B_{m,n}^{n,k}$ such that $B_{m,n}^{n,k}$ is made of all the pixels $\mathbf{p}$ whose coordinates satisfy the relation

$$m = \lfloor x/B_s \rfloor \text{ and } n = \lfloor y/B_s \rfloor. \quad (3)$$

The value B_s is the maximum block size and, after experimental tests, was set to 4. The SAD threshold is set equal to 1.6 times the average SAD computed on the reconstruction error for Intra frames. In case the SAD is small enough, $R_{n,k}$ is kept as is. Note that since this utter partitioning is optional, the encoder needs to notify it to the decoder. An additional bit for each $R_{n,k}$ is coded in the bit stream signalling whether the superpixel has been splitted into $B_s \times B_s$ blocks or not.

Figure 4 shows a graphic example of this process where $R_{n,k}$ is overpartitioned, while $R_{n,t}$ remains unchanged. In this way, a new partitioning (referenced as $S_n^{M'}$) is obtained where regions are referenced as $R'_{n,k}$.

Figure 5 shows an example of the different partial segmentation layouts related to the different steps of the partitioning routine.

2.2 Motion Estimation

In the previous section we have explained how forward motion estimation is used to find a segmentation of the frames I_n and D_n to be coded from a partitioning found on the previous images $\hat{I}_{n-1}$ and $\hat{D}_{n-1}$. The result of such estimation is a set of motion vectors MV_f that permit to generate a first approximation of I_n (D_n) from the pixels of $\hat{I}_{n-1}$ ($\hat{D}_{n-1}$), as Fig. 3 shows. However, forward motion estimation needs to be refined since some segments could have been overpartitioned and other segments have been enlarged via the dilation operator. As a consequence, it is necessary to refine motion vectors via a backward motion estimation that finds for every region $R'_{n,k}$ in $S_n^{M'}$ a predicting segment in the previous frame that minimizes the cost function defined in eq. (2). Each displacement can be identified by a new set of motion vectors that will be referenced as MV_b in the paper. Note that while the set MV_b aims at identifying the partitioning into superpixels for I_n, the set MV_b aims at reducing the energy of the residual error by refining motion compensation. The whole set of predicting segments constitutes the prediction frames I_n^R and D_n^R that respectively approximates I_n and D_n in the residual coding unit.

2.3 Residual and Entropy Coding

After approximating the current image combining segmentation with motion estimation, the residual signal after temporal prediction

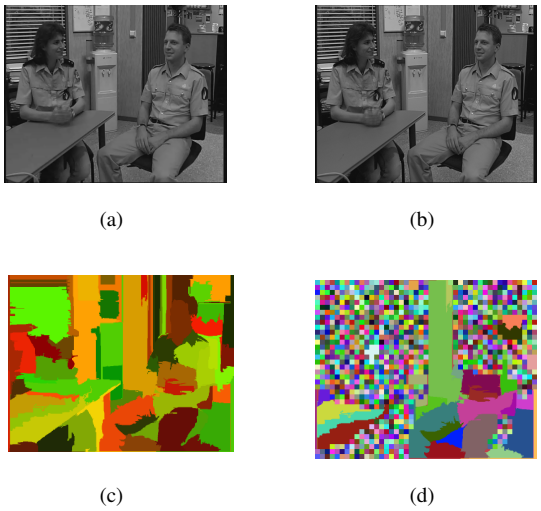

Figure 5: Current and reference images with the corresponding segmentation layouts. a) reference image $\hat{I}_{n-1}$ b) current image I_n c) Segmentation layout S_{n-1} d) Segmentation layout $S_n^{M'}$.

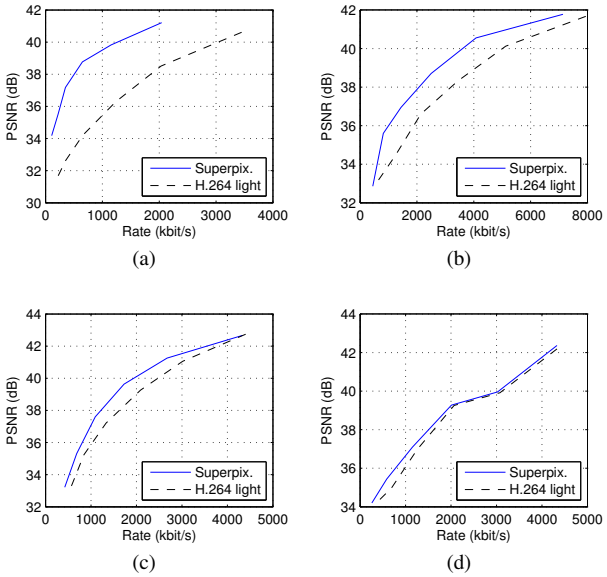

Figure 6: PSNR vs. bit rate of different coding strategies for different sequences (depth component). a) `car` b) `orbi` c) `interview` d) `hands`.

need to be coded. To this purpose, we adopted the coding strategy defined within the H.264/AVC standard [7]. The residual coding strategy is described for the luminance component in the following; a similar strategy is adopted for the prediction residual of depth components. Note that the proposed solution implements separate contexts for syntax elements related to the depth component.

3. EXPERIMENTAL RESULTS

The coder was tested on a wide set of DIBR sequences with different resolutions. We considered sequences `car`, `hands` from Mobile3DTV web site [17], and sequences `Interview` and `Orbi` from [18]. Note that the quality of the provided depth maps changes significantly for the different sequences since different depth estimation algorithms were adopted. Each input sequence is partitioned into GOPs of 15 frames, where the first frame is coded using the standard Intra mode of H.264/AVC [7], while the following frames are coded using the Inter mode. The quantization parameters QP varies within the set $\{20, 24, 28, 32, 36, 40\}$. The chosen entropy coding mode is the CABAC arithmetic coder because of its high compression efficiency and its versatility. CABAC is also employed to code the motion vector sets MV_b and MV_f in the binary stream. The performance of the segmentation-based coder using the segmentation strategy in [16] (labelled `Superpix.`) is compared with a simplified version of H.264/AVC (labelled here as `H.264-light`) which implements predictive mode only and, therefore, proves to be comparable with the proposed architecture since only the transform coefficients and the motion vector set MV_b need to be coded. We also report the coding results obtained with a complete H.264/AVC coder (labelled as `H.264-full`) for the sake of completeness, which implements the main profile with rate-distortion optimization enabled. Figure 6 shows the PSNR-vs-rate curves for different sequences coded with the proposed approach at different bit rates using for segmentation the algorithm in [16] appropriately-modified as described in Section 2.1. It is possible to notice that for the sequence `car`, the segmentation-based method improves the PSNR value of 3 dB at 2 Mbit/s with respect to `H.264-light`, while the improvement increases to 4.5 dB at 7 Mbit/s. A closer look at the bit allocation in the stream shows that 92 % of the bit rate is devoted to coding the residual, while 7 % and 1 % of the bit rate are devoted to coding the sets MV_b and MV_f, respectively. For more static sequences, like `orbi` and `interview`, the compression gain decreases slightly since the difference between the curve of the segmentation-based approach and that of `H.264-light` is about 2 dB for `orbi` and 1 dB for `interview`. This is due to the fact that for static sequences motion compensation performs well using square blocks as well since all the elements present a limited amount of motion and there is no need to distinguish the different objects. It is possible to notice that for extremely-complex sequences (see results for the sequence `hands` in Fig. 6(d)) the performance of the superpixels-based approach does not differ significantly from that of `H.264-light`. This result is partially due to the quality of the estimated depth map, which does not allow an accurate identification of segments in each frame. The plots reported in Figure 7 show the MSE vs. rate curves for the depth component of each subsequences. It is possible to notice that the proposed solution outperforms the `H.264-light` coder.

In our tests, we verified the performance of the algorithm with different segmentation strategies. Fig. 8 reports the experimental results obtained using the k-means approach described in Section 2.1 ((labelled here `Superpix. k-m.`). Experimental results for sequences `car` and `interview` in Fig. 8 show that the improvement brought by the superpixels-based strategy depends on the accuracy of the segmentation algorithm. It is possible to notice that the results obtained using k-means are worse with respect to those obtained via the modified approach by Felzenszwalb and Huttenlocher [16]. The graphs also report the results obtained from a standard H.264/AVC coder (labelled `H.264-full`). In this case, all the rate-distortion optimization options and coding modes are enabled. Note that the superpixels-based approach performs better than the `H.264-Full` coder despite its complexity is significantly lower (coding time saving is about 65 % with respect to `H.264-Full`). As for the `H.264-light` strategy, the coding time of the proposed approach increases of 14 % because of the additional segmentation strategy.

4. CONCLUSIONS

The paper presented a new segmentation-based video coding strategy for DIBR signals. Each frame is partitioned into segments that are to be found in the previous frame via a joint video+depth segmentation of the reconstructed information. Motion vectors are then refined through a re-partitioning and a second stage of back-

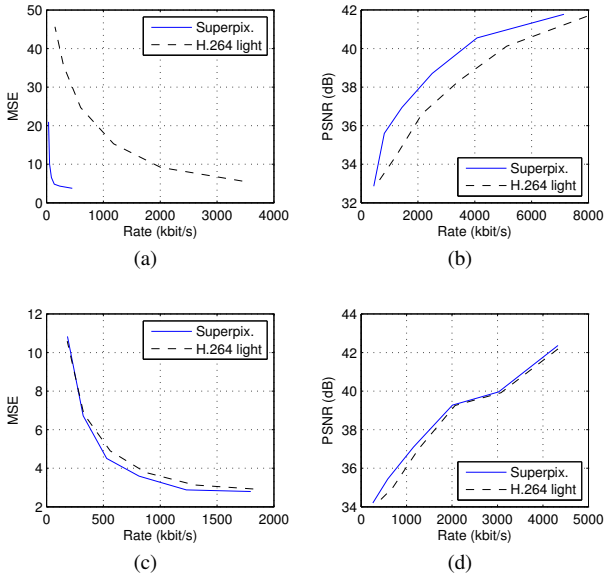

Figure 7: MSE vs. bit rate of different coding strategies for different sequences (depth component). a) `car` b) `orbi` c) `interview` d) `hands`.

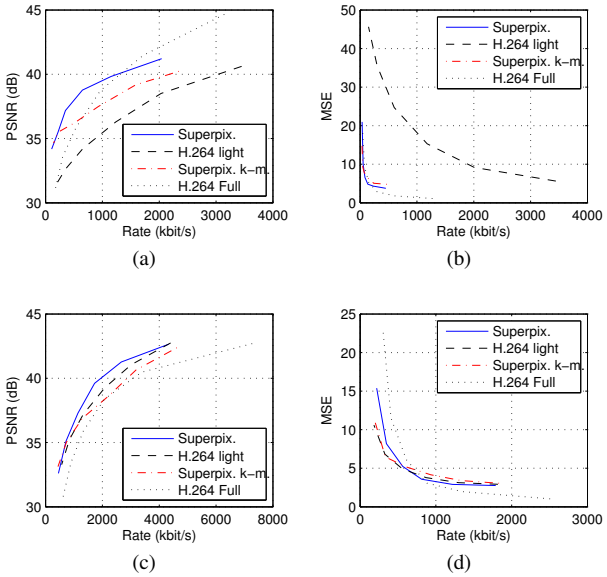

Figure 8: PSNR and MSE vs. bit rate of different sequences for luminance and depth components. a) PSNR for `car` (luminance) b) MSE for `car` (depth) c) PSNR for `interview` (luminance) d) MSE for `interview` (depth).

ward motion estimation. Experimental results show a significant improvement (up to 4.5 dB) with respect to the H.264/AVC standard depending on the amount of motion in the sequence and on the size of the generated segments.

REFERENCES

[1] C. Fehn, "3D-TV Using Depth-Image-Based Rendering (DIBR)," in *Proc. of PCS 2004*, San Francisco, CA, USA, Dec. 2004.

[2] 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.

[3] 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. of IEEE International Conference on Image Processing (ICIP 2007)*, San Antonio, TX, USA, Sept. 16 – 19, 2007, vol. V, pp. 105–108.

[4] Jun Zhang, M.M. Hannuksela, and Houqiang Li, "Joint multiview video plus depth coding," in *Proc. of IEEE ICIP 2010*, Hong Kong, Sept. 26 – 29, 2010, pp. 2865 –2868.

[5] Siping Tao, Ying Chen, Miska M. Hannuksela, Ye kui Wang, Moncef Gabbouj, and Houqiang Li, "Joint texture and depth map video coding based on the scalable extension of h.264/avc," in *Proc. of IEEE ISCAS 2009*, Taipei, Taiwan, May 24 – 27, 2009, vol. 3, pp. 2353 –2356.

[6] I. Daribo, C. Tillier, and B. Pesquet-Popescu, "Motion vector sharing and bit-rate allocation for 3d video-plus-depth coding," *EURASIP J. Appl. Signal Process.: Special Issue on 3DTV*, vol. 2009, no. 3, pp. 3:1 – 3:13, Jan. 2009.

[7] Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, "Joint final committee draft (JFCD) of joint video specification (ITU-T Rec. H.264 — ISO/IEC 14496-10 AVC)," in *Joint Video Team, 4th Meeting*, Klagenfurt, Germany, July 2002.

[8] T. Davies1, K. R. Andersson, R. Sjberg, T. Wiegand, D. Marpe, K. Ugur, J. Ridge, M. Karczewicz, P. Chen, G. Martin-Cocher, K. McCann, W.-J. Han, G. Bjontegaard, and A. Fuldseth, "Suggestion for a test model," in , 1st *Meeting*, Dresden, Germany, Apr. 15 – 23, 2010, files: JCTVC-A033.doc.

[9] P. Salembier, F. Marqués, and M. Pardàs, "Segmentation-Based Video Coding: Temporal Linking and Rate Control," in *Proc. of the EUSIPCO 1996*, Trieste, Italy, Sept. 1996, vol. I, pp. 455 – 458.

[10] F. Marqués, P. Salembier, M. Pardàs, R. Morros, I. Corset, S. Jeannin, B. Marcotegui, and F. Meyer, "A segmentation-based coding system allowing manipulation of objects (Sesame)," in *Proc. of ICIP 1996*, Lausanne, Switzerland, Sept. 1996, vol. III, pp. 679 – 682.

[11] ISO/IEC JTC1, "Coding of Audio-Visual Objects - Part 2: Visual," ISO/IEC 14 496-2 (MPEG-4 Visual version 1), Apr. 1999; Amendment 1 (version 2), Feb. 2000; Amendment 4 (streaming profile), Jan. 2001, Jan. 2001.

[12] C. Zhu, X. Sun, F. Wu, and H. Li, "Video coding with spatio-temporal texture synthesis and edge-based inpainting," in *Multimedia and Expo, 2008 IEEE International Conference on*, Apr. 23 – 26, 2008, pp. 813 –816.

[13] S.B. Gokturk, H. Yalcin, and C. Bamji, "A Time-Of-Flight Depth Sensor - System Description, Issues and Solutions," in *Proc. of CVPRW 2004*, June 27 – July 2, 2004, vol. 3, p. 35.

[14] A. Saxena, M. Sun, and A. Y. Ng, "Make3D: Learning 3D Scene Structure from a Single Still Image," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 5, pp. 824 – 840, May 2009.

[15] R.O. Duda, P.E. Hart, and Stork D.G., *Pattern Classification, 2nd Edition*, Wiley, Nov. 2000.

[16] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient Graph-Based Image Segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167 – 181, Sept. 2004.

[17] Mobile3DTV project, "Repository of Mobile3DTV project: 3D Video database - http://sp.cs.tut.fi/mobile3dtv/stereo-video/," 2011.

[18] Fraunhofer HHI, "Repository of HHI," 2011.