3D coding tools final report

Vincent Ricordel, Vincent Jantet, Josselin Gautier, Christine Guillemot, Laurent Guillo, Emilie Bosc, Fabien Racapé, Luce Morin, Muriel Pressigout, Marco Cagnazzo, et al.

Projet PERSEE
« Schémas Perceptuels et Codage vidéo 2D et 3D »
n° ANR-09-BLAN-0170

Livrable **D4.3** *31/07/2013*

3D coding tools final report

Vincent	RICORDEL	IRCYNN
Vincent	JANTET	IRISA
Josselin	GAUTIER	IRISA
Christine	GUILLEMOT	IRISA
Laurent	GUILLO	IRISA
Olivier	LE MEUR	IRISA
Emilie	BOSC	IETR, IRCYNN
Fabien	RACAPE	IETR, IRISA
Luce	MORIN	IETR
Muriel	PRESSIGOUT	IETR
Marco	CAGNAZZO	LTCI
Giuseppe	VALENZISE	LTCI
Béatrice	PESQUET-POPESCU	LTCI

Contents

Introduction

In this document we describe the latest contributions to the PERSEE project related to 3D video compression. Several perceptually-oriented appoaches have been proposed to encode depth information. Several of these algorithms have been integrated with the new 3D-HEVC 3D video codec which is being developped in the ISO/ITU JVT joint standardization group.

1 Edge based Depth Map Compression for 3D Coding

1.1 Introduction

At the coding stage along the 3DTV framework, the depth map compression methods have recently gain interest in the video coding community. The transmission of geometry joined to the texture information is necessary to Depth Based Image Rendering (DIBR) for interactive multi-view display on 3DTVs. The geometry provided by the depth of the acquired scene will allow the generation of new virtual viewpoints adapted to the end-user's display requirements: additional viewpoints inside or outside of the range of transmitted views to offer more views and then extend the range of stereoscopic vision on auto-stereoscopic displays for example.

It is indeed essential to efficiently encode and reconstruct a depth map in a way that preserves the distance properties of objects within each other because this will be precisely used for later view synthesis. In contrast the depth map smoothly varying surface leads to very predictable pixel values along the spatial dimension that might not need to be perfectly reconstructed if they don't affect the rendering.

A depth map is basically representing different depths of objects in the scene and more exactly different distances to object surfaces. These surfaces are delimited by the object borders. A distortion into the object depth might result in a local deformation of the rendered object, but a distortion at borders between the object and its background depths might lead to worse artefacts: mixed foreground/background object textures.

In this section we will present an efficient depth map compression introducing lossless edge coding as an alternative to 3DVC depth map coding methods approximating piecewise linear functions whose coefficients need to be encoded.

1.2 A Lossless Edge based Depth Map Coding Method

Depth maps have two main features that must be preserved but can also be relied on for efficient compression. The first one is the sharpness of edges, located at the border between object depths. Distortions on edges during the encoding step would cause highly visible degradations on the synthesized views, that may require depth map post-processing. The second one comes from the general smooth surface properties of objects whose depth is measured.

While Merkle et al. first proposed that smooth regions could be approximated by piecewise-linear functions separated by straight lines, we indeed assume that these smooth surfaces could be entirely reconstructed by interpolating the luminance values from their boundaries. Then the coefficients of the piecewise-linear functions would not be transmitted but instead the pixel values on both side of the edges.

To this end, we can observe that depth maps share similarity to cartoon-images. Mainberger et al. [33] proposed a dedicated cartoon-image encoder, that - in low bitrate conditions - beats the JPEG-2000 standard. After a Canny edge detection, the edge locations are encoded with a lossless bi-level encoder, and the adjacent edge pixel values are lossy quantized and subsampled. At the decoding stage, a homogeneous

diffusion or an interpolation are used to retrieve the inside unknown areas from lossy decoded edges. Indeed, the demonstrated performances -while beating state of the art codecs- reach the limit of 30dB.

We revisited this edge-based compression method by proposing improvements to fit the high quality, low bitrate, and specific requirements of depth maps. Finally, we increase the diffusion-based depth map encoding performance, which might be generalized to all kinds of images.

In the next sections the encoding process is described. Then the decoding, diffusion and interpolation methods are explained. Results, performances and comparison with state-of-the-art methods based on a traditional objective metric are then given in the next section, before subjective experiments draw more meaningful results on the effectiveness of our proposal.

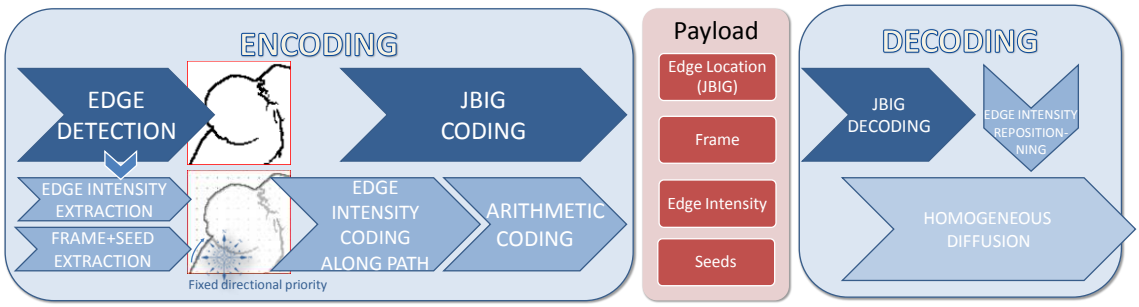


Figure 1: Diagram of the proposed depth map compression method.

1.2.1 Encoding

The encoding is a 5-step process: first is the detection of edges, then encoding of the edge location and finally encoding of the edge, border and seed pixel values, as illustrated by Figure 1.

Edge detection Different operators exist to extract the contour of an image. An optimal edge detector should provide:

- a good detection: the algorithm should find as much real edges as possible.
- a good localization: the edges should be marked as edges as close as possible to the real edges.
- a good robustness: as much as possible, the detector should be insensitive to noise.

In our context of depth map edge coding, several requirements are added. The quality of reconstruction by diffusion should be maximized, while minimizing the number of edges to encode. To avoid diffusion from bad positioned edges causing "leakages", the localization of contours should be quasi-perfect. The detection of contours should be good but avoiding an over-detection. Up to a certain limit, weak contours (i.e. with a low gradient) might be useless for the reconstruction and might unnecessarily increase the edge coding cost. Also, noisily detected pixels should be avoided for the same reason.

The Marr-Hildreth edge detector combined with Canny-like hysteresis thresholding is used in [33], but suffers from errors of localization at curved edges. The widely used Canny edge detector has also been benchmarked. It relies on a 5x5 gradient prefiltering to cope with noise before local maxima edge detection. But this prefiltering step also makes this detector vulnerable to contour localization errors, as illustrated in Figure 2(c), where inexact selection of adjacent edge pixels leads to improper diffusion. In contrast Sobel has the advantage of an accurate contour localization -as shown in Figure 2(d)- at the cost of a noisy, edge over-detection. To cope with these over-detected edges, contours c shorter than a certain value ($c < 14$) are excluded. Pixels with a bi-dimensional gradient amplitude larger than a threshold λ are extracted. Used with sharp depth maps, this gives well-localized contours.

Encoding the contour location As in [33], a bi-level edge image containing the exact location of previously detected edges is first encoded using the *JBIG (Joint Bi-level Image Experts Group)* standard. This is a context-based arithmetic encoder enabling lossless compression of bi-level images. We use the JBIG-Kit, a free C implementation of the JBIG encoder and decoder. The progressive mode is disabled to reduce the required bitrate.

Encoding the contour values Once the edge pixel locations have been encoded, the pixel luminance values have also to be losslessly encoded following our initial requirements. The authors in [33] proposed to store the pixel values on both sides of the edge, instead of the pixel values lying on the edge itself. Indeed, for blurry contours, this might be valuable to interpolate the inner part of the edge and code the luminance values on both sides. However, with sharp depth maps, the pixel values lying directly on an edge, as illustrated in Figure 2(b), alternate between one side or another from this edge and couldn't be interpolated correctly.

With the Sobel edge detection not thinned to a single edge pixel, we ensure retaining at least one pixel value from each side of the frontier edge as shown in Figure 2(d).

We keep the idea of storing the pixel values by their order of occurrence along the edge to minimize signal entropy. A path with fixed directional priorities (E, S, W, N, NE, SE, SW and NW) is used. As the intrinsic properties of pixels along an edge or "isophote" are their small luminance variation, then we propose to compute the differential values of edge pixels in a Differential Pulse Code Modulation (DPCM) way. From this optimized path encoding method, the stream of DPCM values is then encoded with an arithmetic coder.

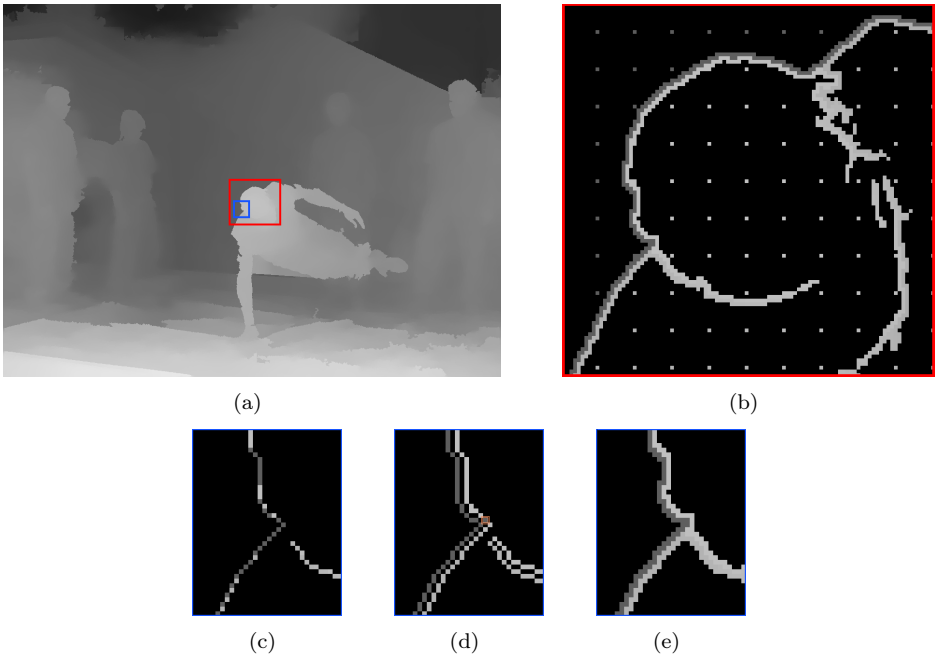


Figure 2: (a) A "Breakdancer" depth map, (b) the encoded and decoded Sobel edge and seed pixels (red selection on (a)), (c) a zoom (blue selection) on Canny edges, (d) the selection of corresponding pixel adjacent to Canny edges (c) as in [33], with an intruder edge pixel (orange-framed) that will lead to bad diffusion, (e) the proposed Sobel selection of edge pixel values, exactly located from both side of the frontier edge.

Additionally to these edges we also encode two kinds of information. The pixel values from the image border are stored to initiate the diffusion-based filling from borders. Inspired by the work of [4] on "dithering" for finding optimal data for interpolation, we propose to sparsely deploy, at regular intervals, some seeds of original depth pixels as shown in Figure 2(b) (The interval $s = 10$ in practice) While having low overhead, we discovered that this helps accurate reconstruction by initializing and accelerating the diffusion in large missing areas.

Thus, these extra border and seed pixels are coded in DPCM and added to the differential edge values. The resulting file is thus composed of the payload of the JBIG data and of the arithmetic encoded bitstream of the DPCM border, edge and seed pixel values. A typical PCM payload and its subsequent DPCM payload (from a depth map encoding of the "breakdancers" sequence) are illustrated in figure 3.

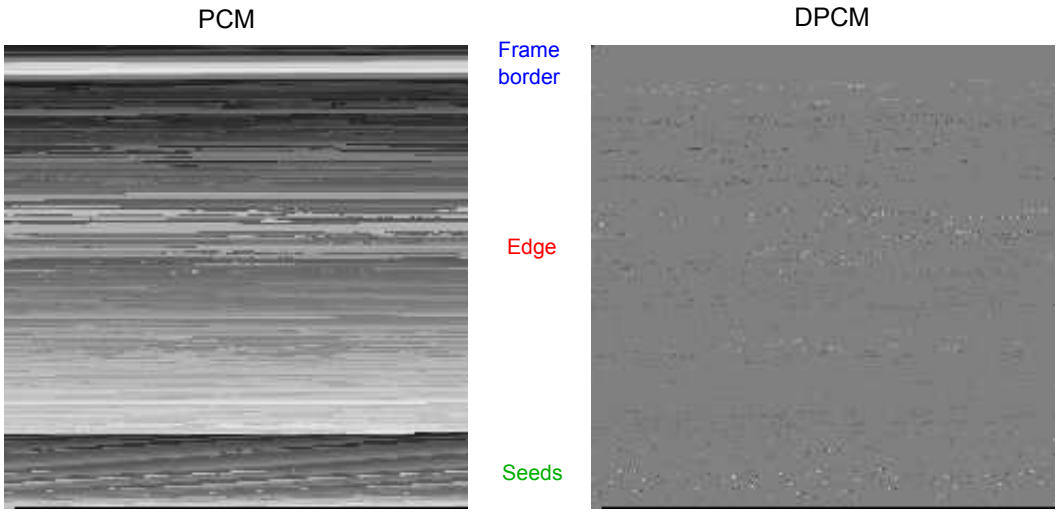


Figure 3: Illustration of the payload of the encoded pixels intensity values ("breakdancers" depth map). The PCM (left) pixel intensity and the DPCM (right) residuals are displayed in raster scan order.

It can be seen on the PCM values (3 left) that the pixels vary very smoothly along the frame border: the frame border pixel entropy is very low after DPCM on upper part of 3 right. Interestingly , the upper frame border is black while the lower is bright: it comes from the scene configuration: the upper part is the farther part, the background whole, and lower part of the scene is the ground closer to the camera.

The original pixel values of the edge pixels coded with a walk along the edge appear more cluttered: edges pixels contribute the most to the global entropy of the image. The edges are found and coded in a raster -scan order on the whole image: even if the edges have low-varying values along their edges, once an edge is terminated, the iterative process follow the encoding from the next vertex whose values can be very

different. Each jump to the next edge can be illustrated by the higher or lower residual error than the middle gray value in 3 right.

Finally the seeds values appear regular: their luminance is encoded in a raster scan order so the highest residual pixels come from a jump from seeds on background to foreground and reversely.

1.2.2 Decoding, diffusion vs interpolation

A lossless decoding of the border, edges and seeds is performed. Then two methods are proposed and evaluated to reconstruct the depth map surfaces: a lossy diffusion or an interpolation from the decoded edge pixels. Finally, a quadtree approach is proposed to place the seeds.

Decoding contour location and pixel values Once the edge positions from JBIG payload are decoded, the edge pixel values are decoded and positioned following the same order in which they were encoded: the path along contour location respecting directional priorities. The border and seed values are also re-positioned following a predefined location.

Reconstructing the Missing Values by Diffusion We now have a sparse depth map containing only the edge, border and seed pixel values. A homogeneous diffusion-based inpainting approach is used to interpolate the missing data. This method is the simplest of the partial differential equations (PDEs) diffusion method, and has the advantage of low computational complexity. It directly stems from the heat equation:

$$\begin{cases} I_{t=0} = \widetilde{I} \\ \frac{\delta I}{\delta t} = \Delta I \end{cases}$$

where $\widetilde{I}$ is the decoded edge image before diffusion that will constitute the Dirichlet boundaries of the equation. The diffused data then satisfies the Laplace equation $\Delta I = 0$. The diffusion process is run in a hierarchical manner, each diffusion step being in addition helped with seeds and appropriate initialization. These three improvements have been introduced in the classical diffusion approach to limit the number of iterations required to converge, hence to speed up the entire process.

diffusion A Gaussian pyramid is built from $\widetilde{I}$. The diffusion process is first performed on a lower level of the pyramid and the diffused values are then propagated to a higher level (3 levels are used and show good performances). The propagation of the blurred version of the diffused pixel values from a lower level to an upper one helps to initialize the diffusion in unknown areas.

Middle range initialization On the highest level, instead of starting from unknown values of $\widetilde{I}$ set at 0, we propose to initialize unknown values to the half of the possible range: 128 for an 8 bit depth map. This facilitates and speeds up the process of diffusion by limiting the number of required iterations to converge.

Seeding As explained in section 1.2.1, some seeds are chosen from a regular pattern both to accelerate the diffusion process and to provide accurate initialized values in large unknown areas. Indeed, this definitely achieves a fast and accurate diffusion -with a gain of 10 dB- for a quasi-exact reconstruction of the depth map.

Reconstructing the Missing Values by Interpolation A bi-linear interpolation is also tested to render the missing values between border, edges and seeds. A linear interpolation in both directions from the border, edge and seed pixels is simply realized. It is equivalent to a two-table lookup with linear interpolation in both horizontal and vertical directions.

This approach is more simpler than the diffusion one and gives better results: no iterative process of diffusion are required up to an asymptotic point. This will be assessed in the section 1.3.

1.2.3 An Extension by a Quadtree Approach

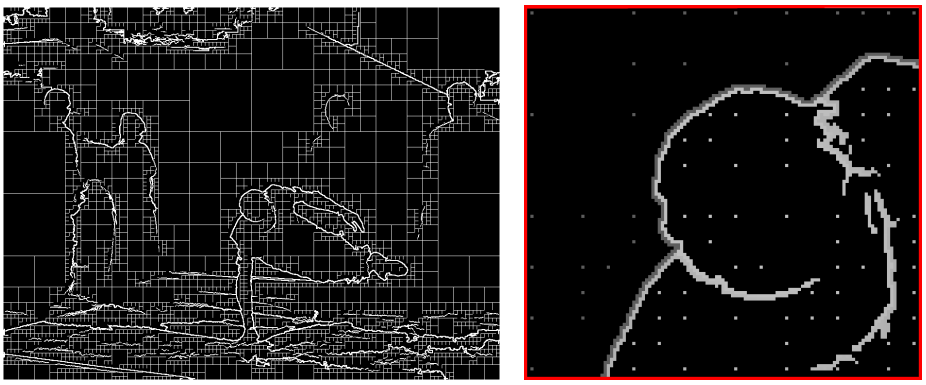
Another approach has been finally proposed to add flexibility to the coding of seeds. Instead of placing the seeds on a regular pattern, the seeds are positioned on each vertex of blocks obtained from a quadtree decomposition. The original depth map is recursively divided into four equal-sized square blocks based on the presence of edges within the block. If an edge is present within a given block, this block is subdivided into smaller blocks, up to the limit of the smaller size of block.

Because the edge positions are transmitted and the smaller size of block is fixed and known at the decoder side, no quadtree decomposition needs to be conveyed: only the border, edge, and seed -positioned according to the quadtree- values are transmitted in addition to the pixel location.

A quadtree decomposition of a "breakdancers" depth map is illustrated in 4(a). The quadtree is well decomposed into smaller blocks along the edges, while big blocks remain in the large smooth areas. However, the interpolation in these areas might be poor because only four seeds at vertices will help the interpolation (except at frame border). Small variation of depth surfaces not extracted by edge detector will not be reproduced on large areas, while they would be partially reproduced with seeds placed on a regular pattern. This will be assessed in the next section. The hypothetical advantage to use seeds placed according to a quadtree rather than on a regular interval is illustrated on Figure 4(b), to compare with 2(b). The seeds are more dense in the neighbourhood of detected edges which might lead to a better interpolation and reconstruction of the small surfaces.

1.3 Results based on objective quality evaluation

The performances of the proposed compression method and its extensions are first evaluated on an objective quality metric ground. An original resolution depth map from a MVD sequence "Breakdancers" from [**?**] is used. It was accurately estimated through a color segmentation algorithm. The original good quality of the depth map



(a) (b)

Figure 4: Ilustration of the quadtree decomposition on the criterion of edge presence. (a): Quadtree decomposition blocks are illustrated in white. The minimum size of block is 8x8. (b) A zoom on 2(a) with the seeds positioned according to the quadtree.

will enable a precise evaluation of the impact on texture synthesis reconstruction of the depth distortions.

1.3.1 Depth map objective quality evaluation

The reconstruction quality of our PDE-diffusion-based method is investigated and compared with the JPEG-2000 and HEVC-HM-4.1 Intra compressed versions. First, to visually illustrate the difference of quality reconstruction on edges, the three methods are compared at equal *Peak-Signal-to-Noise-Ratio* (PSNR),(45 dB, JPEG-2000 with a Quality factor Q=25, HEVC-Q=40).

A zoom on the head of a character presenting initially sharp edges highlights the difference of edge quality depending on the compression type (Figure 5). While at high PSNR, the JPEG-2000 (a) and HEVC (b) versions of the depth map tend to blur the edges. This is commonly referred to as ringing artifacts. It appears with JPEG-2000 because of the lossy quantization following wavelet transformation. It might appear with HEVC because of deblocking filter limitation. Then both JPEG-2000 and HEVC cannot efficiently reconstitute the smooth gradient on uniform areas while preserving the edges. In contrast, our proposed approach stores the exact edges and diffuses regions between these edges, resulting in a smooth gradient restitution on slanted surfaces and non-distorted edges.

Thus we evaluate the global depth-map rate-distortion performances of the three previous encoding methods plus the interpolation and the interpolation-plus-quadtree methods. Figure 6 shows that the diffusion approach outperforms JPEG-2000 except in very low or high bitrate conditions, while being under HEVC. No dedicated adjustment

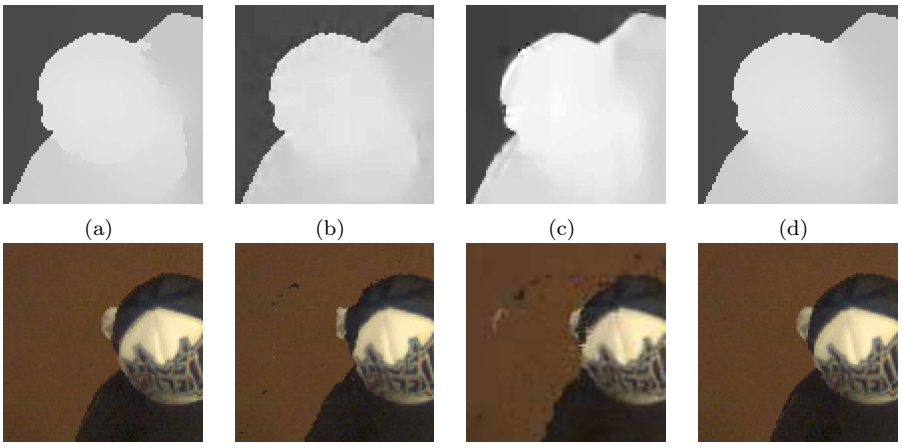


Figure 5: Upper row: zoom on the head of a dancer on original View #3 (V_3) depth map (a) highlights -by comparison at equal depth map PSNR (45dB) referenced to (a)- the ringing artifact on JPEG-2000 (b) and the blur effect with HEVC (c). Our method (d) based on exact edges and homogeneous diffusion prevents this effect (contrast has been increased on depth maps for distortion visibility). Lower row: zoom on corresponding synthesized view V_4 without (e) or with JPEG-2000 (f) and HEVC (g) compressions and our diffusion-based method (h).

was performed in our method, only the threshold λ was varied to adjust its bitrate (an interval of 10 pixels between seeds was chosen for the tests).

The interpolation approach with the same regular pattern of seeds gives substantial improvements in terms of depth map PSNR (with the original depth map as reference). An average gain of 4dB in term of PSNR is obtained in practice between the interpolated-decoded depth map and the diffused-decoded depth map.

With the quadtree approach, the supposed advantage of reduced seed number on the bitrate is counterbalance by the decrease of depth map PSNR quality. For different minimum sizes of quadtree blocks, 8 and 32 pixels, the average fall of quality is of 6dB and 7.5dB on average respectively. The maximum size of quadtree blocks was 512 pixels and might also have been limited to a small size. As proposed before, it is precisely on these large uniform areas that the loss of PSNR quality is important. But have these falls repercursions on the objective visual quality of the synthesized - and then displayed - view? This will be presented in the next section.

1.4 View synthesis quality evaluation

The impact of depth compression methods on rendering is measured by calculating the PSNR of a synthesized view (from a pair of uncompressed textures and compressed depth maps), with respect to an original synthesized view (from a pair of uncompressed textures and depth maps). The corresponding synthesized view from two original

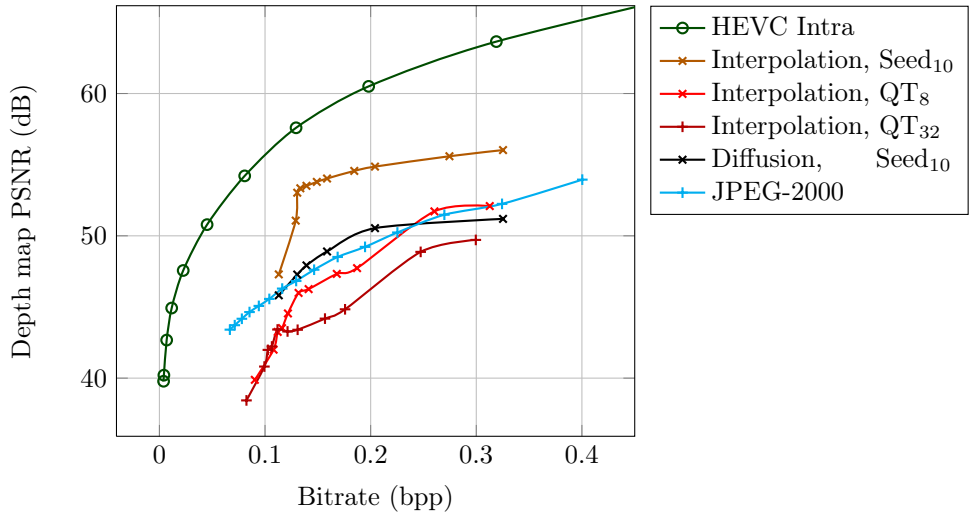


Figure 6: Rate-Distortion performance of the V_3 first "breakdancers" depth map with different quality factors of JPEG-2000 and HEVC and different Sobel detection thresholds λ for the three proposed methods. The proposed methods differ in the interpolation from decoded edges and seeds: diffusion or bi-linear interpolation from regular seeds and bi-linear interpolation from quadtree-placed seeds.

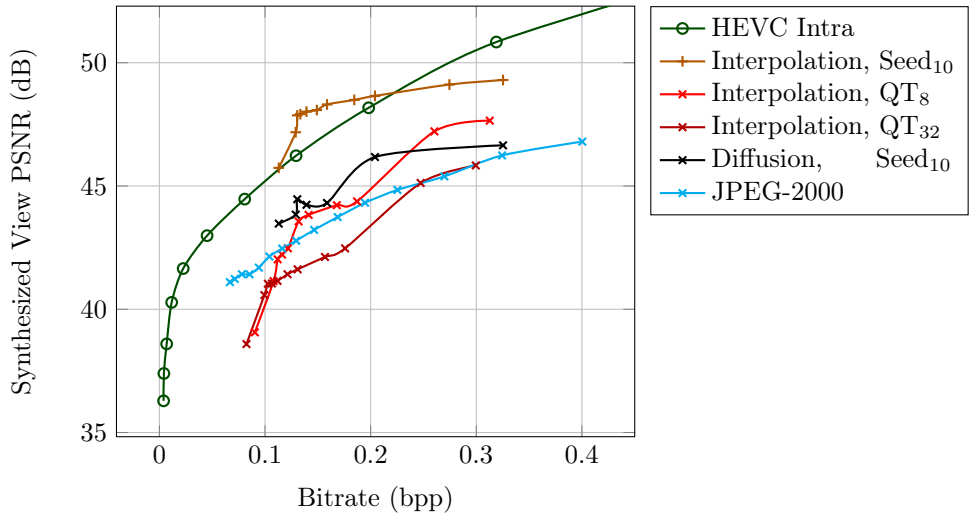


Figure 7: Rate-Distortion performance of synthesized V_4 with the bitrate of V_3 depth map, for different quality factors of JPEG2000 and HEVC, and different Sobel detection thresholds λ for the three proposed methods.

depth maps is then the reference. VSRS 3.0 [32] is used for view interpolation from this 2-view dataset.

The R-D synthesis performance, illustrated in Figure 7, justifies the edge-coding approach over wavelet based encoders: undistorted edges permit an accurate and efficient view coding and rendering. The PSNR quality of synthesized view is better than JPEG-2000 with the edge-based method with both diffusion or interpolation filling from regular seeds. The interpolation method even beats the HEVC intra coded method for 0.1 to 0.2 bpp. However, the PSNR measure shows its limitation of objective evaluation on perceived quality. Our method does not always outperform in term of rate-distorsion the existing methods (Figures 6, 7), but still can improve the perceived quality of the synthesized view, especially around critical edges (see Figure 5).

The non-linearity of the depth map and synthesized view PSNR quality deserves some explanation. The rate-distorsion PSNR curves are all monotonic but a drop appears with diffusion-based and especially with interpolation-based solutions when the bitrate is reducing under 0.12-0.14 bpp. This can not be explained by the constant cost of seeds along a regular pattern that could become not negligible at low bitrate. The quadtree solution -which effectively reduced the cost of seeds and then the total cost of the methods- still introduces this non-linearity.

This fall might be due to detected edges non-connected anymore. When reducing the bitrate and then the edge detection threshold, the edges around object become non-connected or their edge pixels values along the edge are not well preserved. In those cases with our walk-along-edge technique, when a hole around an edge is encountered, a new vertex non-correlated with the preceding edge pixel has to be transmitted. When this happens over the whole image, the total quality of depth map reconstruction is affected because entire surfaces become poorly interpolated. A solution could be to reconnect the edge pixels at decoding side before interpolating their values along the edges and then between the edges.

1.5 Subjective Results

Subjective assessment of the influence and impact of compressed depth maps on 3D view synthesis have been conducted in the IVC lab of IRRCyN in Nantes. The experiments were in line with the MPEG on-going activities on 3D video coding and then relied on similar test conditions and view rendering techniques and the same sequences as input tested data. These were realized within the PERSEE project context and involved different depth coding methods of four french labs.

The idea was to observe, test and measure the influence of the proposed depth compression method on the perceived visual artefacts of rendered views and then on the consequent perceived visual quality by observers.

1.5.1 Experimental contents

The first experiment presented below involves the rendering of multiple spatially-translated but temporally-fixed intermediate frames from the same two adjacent views obtained from uncompressed textures and compressed depth images. An intermediate rendering from a pair of frames was processed, then another shifted in space rendering from the same pair was processed and so on. The resulting shifted intermediate frames were then concatenated into a single monoscopic video displayed on a monoscopic screen. This leads to a video where the viewpoint is changing over time while the acquired time doesn't evolve: a bullet-time effect video.

Depth Map Coding Methods Two state-of-the-art simulcast and one multiview-plus-depth video coding methods are selected for the benchmark: H.264/MPEG-4 AVC, HEVC and 3DVC respectively. The depth coding method is also compared to the recent JPEG-2000 still image coding standard which is based on wavelet-transform and is then supposed to efficiently encode the 8-bits uniform area while limiting the ringing artefacts. The selected method from our previous work was based on interpolation and quadtree seed distribution with a quadtree block minimum size of 32 pixels. This choice was made before the objective quality comparison with previous interpolation methods presented in 1.4. This method was originally selected according to a bitrate criterion to compete with the state-of-the-art 3DVC technique. Finally, filtered and edge-filtered versions of the original depth map are tested for extensive comparisons.

Preselection of Coded Depth Map Versions According to the Perceived Synthesized Quality Because the objective quality scores -with their associated bitrate- of depth maps could not give a good opinion on the range of subjective quality of the synthesized views, three experts first pre-selected different qualities of coded depth maps according to three classes of perceived quality of synthesis. This to ensure that the rank were given in roughly the same range of quality. Then the quality of synthesized views could be compared between each other into a dedicated class and evaluated by an observer.

Competing with 3D-HTM The main issue when we want to compare a method to another one is how to do that fairly. This issue arises with the comparison of our depth map coding technique to the depth coding method embedded into the 3DVC video standard.

First, while our method does not involve any temporal prediction, it seems fair to compare the rate of the depth map coding technique to the distortion in view synthesis, expressed on a single frame instead of on the whole video. The intra mode of H.264 and HEVC are then used for the single frame comparison. The bitrate of the single Intra coded depth image is then used, while the distortions are evaluated on the resulting rendered temporally-shifted video.

Second, while our method uses in input only a single depth map, the 3DVC depth map coding uses a set of different inter-component prediction techniques to efficiently encode jointly the texture and the depth data (see sections **??** and **??**), such as the inter-prediction Mode 3 and 4, but also the View Synthesis Prediction.

View Synthesis Method The view synthesis method is the same as used by MPEG: the View Synthesis Rendering Software (VSRS). The same software version as currently used for normalization of MPEG 3D-HTM is retained, and the same rendering is used: view interpolation from two adjacent views. Two modes of interpolation are used, either with view blending or without.

The view synthesis is run with a pair of non-compressed texture images and the corresponding pair of compressed depth maps coded with one of the depth coding methods. Then, the impact of the depth map coding is isolated and can be measured independent of texture coding.

Resulting bullet-time video stimuli All the tested methods are compared at three levels of quality of the resulting synthesized view, predefined by a video quality expert. For each depth map coding the blending mode of VSRS is either activated or deactivated. Thus, for each method, 3 levels x 2 blending modes are tested, so 6 bullet-time video stimuli were displayed, evaluated and annotated by the viewer.

Between each frame, the intermediate view was shifted $^1/_{50}$ from the left toward the right view. When the extreme intermediate right view before the original right view was synthesized, the inverse camera movement was done. Then, 50 resulting frames shifted toward the right plus 50 frames in inverse movement were displayed successively. This allows the viewer to clearly identify the potential artefacts appearing in a moving video -as it will be the case in practice- while the pair of depth maps is the same over time.

Test material Six videos as proposed by MPEG have been retained. Two Class-A Full HD video and four Class-C 1024x768 pixels resolution videos were used.

Sequence	Encoded views	Displayed views
Undo Dancer	1-9	$(1 +^1 /_{50*(9-1)-2}) \rightarrow (9 -^1 /_{50*(9-1)-2})$
GT Fly	1-9	$(1 +^1 /_{50*(9-1)-2}) \rightarrow (9 -^1 /_{50*(9-1)-2})$
Kendo	1-5	$(1 +^1 /_{50*(5-1)-2}) \rightarrow (5 -^1 /_{50*(5-1)-2})$
Balloons	1-5	$(1 +^1 /_{50*(5-1)-2}) \rightarrow (5 -^1 /_{50*(5-1)-2})$
Newspaper	2-6	$(1 +^1 /_{50*(5-1)-2}) \rightarrow (5 -^1 /_{50*(5-1)-2})$
Book Arrival	6-10	$(1 +^1 /_{50*(5-1)-2}) \rightarrow (5 -^1 /_{50*(5-1)-2})$

Table 1: Selected multi-view video sequences with their respective encoded and displayed views.

1.5.2 Viewing conditions

The assessment of the quality of the synthesized view was realized on a 2D conventional LCD display (Panasonic BT-3DL2550) in a controlled environment (following the recommendations ITU-R BT500-11). For the HD sequences, the observation distance was of 3H (ratio between height of the screen (310mm) and observation distance (93cm)).

1.5.3 Participants

27 subjects with normal or corrected-to-normal vision participated in the experiment. The experiment was split into two sessions. Each subject evaluated all the video stimuli.

1.5.4 Test protocol

The subjective assessment was conducted with the Absolute Category Rating with Hidden Reference (ACR-HR) methodology, as set forth in ITU-T recommendation P.910. The test sequences were presented one at a time and rated independently on a category scale. Among the displayed stimuli, a hidden reference was included to prevent the assessment values from being affected by differences in the video content used for assessment. The assessment results obtained by the ACR method are "normalized" by using the following formula to calculate the difference in scores between the assessment video and reference video, expressed as DMOS (Differential Mean Opinion Scores):

$$\text{DMOS} = \{\text{assessment video score}\} - \{\text{reference video score}\} + 5$$

with the quality of the reference video judged to be from "1: Bad" to "5: Excellent" by the subject between each video presentation.

1.5.5 Results: DMOS

The average DMOS of the 27 observers for the "Balloons" and "Book Arrival" bullet-time video synthesized sequences are illustrated in Figures 8. Additional results for the rest of MPEG sequences of 3DVC corpus are presented in Annex **??**. Different trends can be observed for the set and for each video sequence. These will be presented before the limitations and perspectives be discussed.

Globally speaking, the evolution of DMOS appears similar between the versions of VSRS rendering with or without blending, and this whatever the sequences.

Also, the 3DVC Intra coding of depth maps generally performs best among the low quality class of video. However, the 3DVC intra coded methods sometimes also present by far the lower subjective quality (Balloons and Kendo not blended). This raises the question of the best **trade-off** between **perceived quality** and **bitrate requirements**.

The compressions of the five methods selected in the middle quality class gives globally a subjective score which is inside the confidence interval: the five methods manage to compress effectively the sequence in a relatively low range of bitrate without affecting the perceived visual quality (between 0.025 and 0.04 bpp for class C videos). These two observations are important because they tend to show that -without considering the texture compression- a good compromise can be found between perceived quality and bitrate up to a certain limit. Under this limit, the bitrate is effectively decreased but so is the quality.

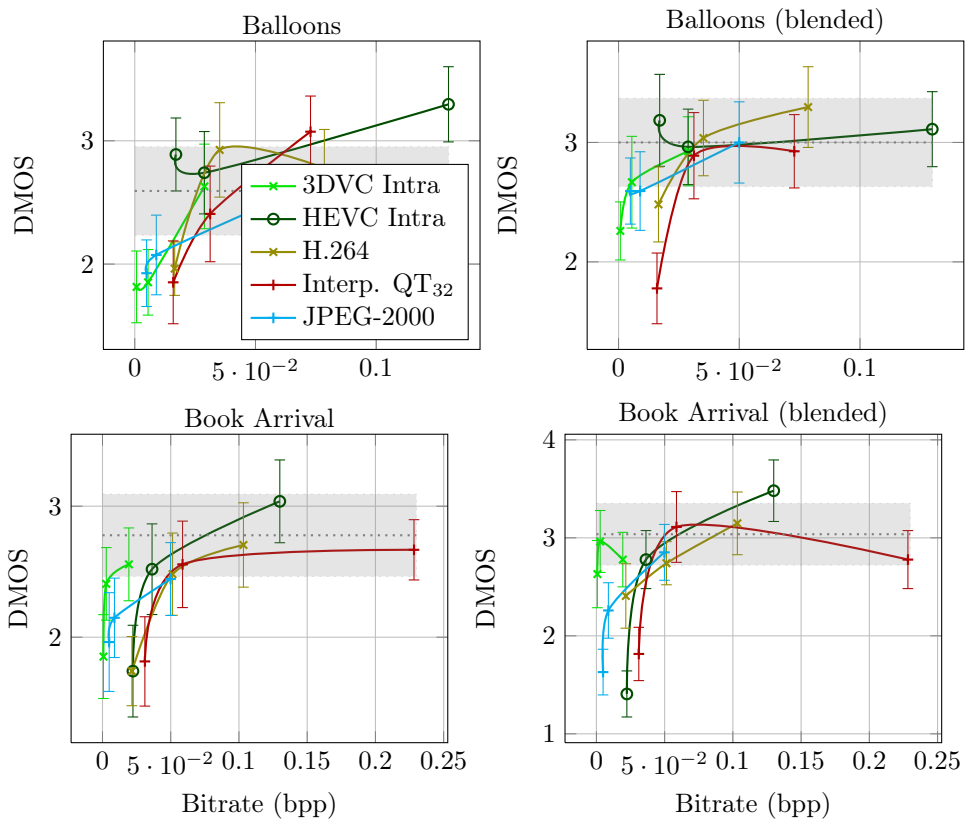


Figure 8: Average DMOS reported by 27 observers on Class-C "Balloons" and "Book Arrival" bullet-time synthesized videos for 3 classes of quality (high, middle and low quality) with 2 modes of rendering by the VSRS interpolation software (blending on/off to the right/left columns respectively) and for five depth map encoding methods. The DMOSs are obtained from a normalisation of the viewers' MOS by the MOS of the hidden non-coded reference picture. This HR-MOS is overlaid on the figure with a gray dashed line surrounded by its confidence interval illustrated by a gray area.

It is hard to conclude on the best solution for high quality reconstruction. H.264, lossless-edge coding method (with seeds placed at vertex of block of 32*32 minimum size) and HEVC manage to increase the perceived visual quality over the original depth map, but among this high quality class, their bitrates are often important. This interesting property of increase of quality from the depth map compression might be hard to use in practice because of the joint texture compression and because of the very low allocated depth bitrates.

Concerning each sequence, the "Balloons" sequence DMOS (without blending) of the different methods appears close to the HR-MOS for the middle quality class, but our method exceeds the others in term of rate-distortion ratio only for the high quality class.

On the "Book Arrival" sequence our method suffers from high bitrate at high and low quality class of videos. It exceeds the other method for the middle class in blended mode, but not significantly.

The "Kendo" is known to have poor quality depth map, so most of the method performances remain in the confidence interval of the original HR-MOS (without blending). The blending deeply increases the perceived video quality of the original reference, so that its performance and confidence interval are higher and lower respectively. Our method does not perform well in this setting for all classes of quality.

The high quality "Kendo" and "Newspaper" compressed with our method show either lower or higher bitrates respectively than the other methods. It shows the limit of the approach. No linear regressions were implemented to limit the size of the depth map according to a bitrate range rather than to an edge detection threshold.

The Full-HD Class A videos are synthetic videos and it seems hard to conclude on a global tendency. The artefacts appeared difficult to evaluate on the "GT Fly" sequences because most of the methods are within the confidence intervals of the original HR-MOS. It means either that the distortions are not visible enough, that the baseline between cameras is too small or that all the methods perform well on this content. This last hypothesis is very unlikely however.

In contrast, the "Undo Dancer" compressed versions are much lower than the hidden original version. This might be explained by the numerous planes at various depth that make the distortions between views very noticeable. Except with 3DVC, all the methods show a DMOS score under the rank of 2 points. Another experiment (with reduced camera baseline, slower movement, etc..) might help to clarify the performance of the five tested methods on this sequence.

Discussion First, the relative small amplitude and high confidence intervals of DMOS for the different tested methods limit the capabilities of interpretations and differentiations of the method performances. Second, the fact that these DMOS are most of the time inside the confidence intervals of the Hidden Reference MOS (HR-MOS) also affects the possible conclusions on the supposed impact of depth compression on perceived quality in 2D. The tested methods have close subjective quality to the original reference, but it seems hard to differentiate them.

However, two clear trends appear. The 3DVC depth map coding gives globally the best "rate-subjective-distortion" performances for the class of low quality, with the lower bitrate but also with the lower DMOS. To evaluate precisely these implications, additional tests must be realized. But it is very likely that in practice the allocated depth bitrate will imply these distortion effects. However, the distortions impacting the perceived quality might be limited by the View Synthesis Optimization.

Also, the impact of the blending on the perceived quality is negligible except for "Kendo" whose original depth maps are very distorted. Additional tests with joint depth and texture compression could confirm or deny the limited impact of blending.

It seems very important to precisely adjust the baseline of views for synthesized videos by pre-tests. This has been done however and baselines were doubled for nearly all videos following an initial rendering. According to the results, the "GT Fly" rendering introduces a too short baseline while the "Undo dancer" baseline configuration is too large. Typically, a large baseline induces large disoccluded areas for each view before merging. It is on these merged areas that potential artifacts will appear. From one sequence to another, the size of these regions might vary, and influence the marks by the viewers. In other words, viewers' marks for a given sequence might be influenced by another preceding sequence and its synthesized quality.

Two important findings are highlighted from this first campaign of experiments in 2D conditions. First, all the methods manage to perform within the same range of perceived quality of synthesis from the original depth map within the middle and high quality classes (see the second and third points along each curves).

Second, and consequently, there seem to exist a **critical threshold** of distortion visibility where the DMOS is dropping, especially for the low quality classes of videos with bitrates under 0.025 bpp. This threshold might be decreased however in 3DVC by the use of advance compression predictions techniques such as the View Synthesis Optimization (VSO).

Perspectives These tests were conducted with the lossless-edge version with the quadtree configuration that led to the worst objective results. The comparison based on an objective metric was not realized before the subjective tests and so the quadtree configuration was retained only according to the low bitrate criterion. In the light of the objective results, the lossless edge encoding method with an **adapted quadtree** could give much better subjective results within the low and middle quality classes of videos, for slightly higher bitrates however. Because additional experiments are planned in stereoscopic conditions, this quadtree configuration giving the best objective scores will be tested.

The experiments raise the question of the best trade-off between quality and bitrate allocated to the depth map. Up to which **limit** can we decrease the depth map bitrate without affecting the synthesis quality so it remains interesting to transmit the depth maps rather than transmitting supplementary texture views (with or without their depth maps)?

Behind this question, two other fundamental questions are asked. How to determine a **correspondence law** between the quality factors of texture and depth maps, this

depending on the baseline? How to evaluate precisely, objectively and subjectively the **impact of distortions** appearing on the **disoccluded areas** with the interpolation or extrapolation rendering methods? This last issue is tackled in the next chapter with the use of objective quality metrics applied on extrapolated synthesized views and on their reconstructed disoccluded areas.

1.6 Conclusion

An alternative method to the new depth map 3DVC encoding has been presented that consists of separately coding the edge location at the picture level through a binary JBIG arithmetic encoder and the pixel values on both side edges in a predefined directional order.

An extension of this method has been proposed. It relies on the idea of an adaptive placement of seeds denser in the edge neighbourhood. But what is gained around the contours is also lost in large areas without contours. These surfaces become poorly reconstructed and then penalize the final synthesised quality. Then a good trade-off has to be found between a maximum size of block not too large that would decrease the synthesized quality and a minimum size of block not too small that would increase the bitrate without gain of quality.

The subjective results confirm the pertinence of the approach, but also show its limitations at very low quality and very high bitrate when using a non-adapted quadtree approach.

Thus the idea of coding the edge location, its partition by predefined pattern or chain code are both relevant. The 3DVC finally simplifies the edge values on both sides of the edge as constant ones. The complexity of the encoder is instead put on the refinement of the quadtree and of the depth block quantization optimizing an RD criterion on the resulting view synthesis: the view synthesis optimization.

Finally, the depth map coding methods may be substantially improved in the near future by considering the ecological structure of the scene; object based and context based approaches might induce relevant improvements on the depth map coding performances.

2 The "Don't Care Region" paradigm for 3D video coding: latest results

In this section we introduce the concept of "Don't Care Region" (DCR), that can be used for the coding of 3D video (see also D4.2 and D3.4). In our previous work on this subject we mainly presented the basic idea of DCR and outlined a set of possible tools using it in the context of 3D image and video coding. Here we introduce a full system based on DCR. The DCR's are integrated in an implementation of the H.264 video codec, and therefore we are able to provide fully constistent experimental results that confirm the advantages related to the use of the DCR.

2.1 The *Don't Care Region* concept

To enhance visual experience beyond conventional single-camera-captured video, elaborate arrays of closely spaced cameras (e.g., 100 cameras were used in one setup in [19]) are now proposed to capture a scene of interest from multiple viewing angles, so that an observer can interactively choose a specific captured viewpoint as the video is played back in time. If, in addition to texture maps (RGB images), depth maps[1] (per-pixel physical distance between scene objects and the capturing camera) are also acquired, then the observer can synthesize successive intermediate views between two camera-captured views via depth-image-based rendering (DIBR) [35] for smooth view transition, achieving free-viewpoint visual experience [27]. Transmitting both texture and depth maps of multiple viewpoints—a format known as *texture-plus-depth*—from server to client entails a large bit overhead, however. In this contribution, we address the problem of temporal coding of depth maps in texture-plus-depth format for multiview video.

The key observation in our work is that depth maps are not themselves directly viewed, but are only used to provide geometric information of the captured scene for view synthesis at decoder. Thus, as long as the resulting geometric error does not lead to unacceptable synthesized view quality, each depth pixel only needs to be reconstructed coarsely at decoder, e.g., within a defined tolerable range. We first formalize the notion of this tolerable range per depth pixel as *don't care region* (DCR) using a threshold τ, by studying the synthesized view distortion sensitivity to the pixel value. Specifically, if a depth pixel's reconstructed value is inside its defined DCR, then the resulting geometric error will lead to distortion in a targeted synthesized view by no more than τ. Clearly a sensitive depth pixel (e.g., an object boundary pixel whose geometric error will lead to confusion between background and foreground) will have a narrow DCR, and vice versa.

Given per-pixel DCRs, we then modify inter-prediction modes during motion compensation in such a way that, for each pixel of a block, we find the smallest residue that brings the predicted pixel inside DCR. This is different from the conventional approach that aims at reconstructing a fixed ground-truth depth block, and results in

[1] Depth maps can either be estimated via stereo-matching algorithms, or captured directly using time-of-flight cameras [21].

a lower energy of the prediction residuals. SKIP mode is also similarly altered, so that code block of the same location in reference frame is evaluated against DCRs in a target block in the current frame. We implemented our DCR-based motion compensation scheme inside H.264 [61]; our encoded bitstreams remain 100% standard compliant. We show experimentally that our proposed encoding scheme can reduce the bitrate of depth maps coded with baseline H.264 by over 28%.

In the following, e first discuss related work in Section 2.2. We then define formally per-pixel DCR in Section 2.3. Given per-pixel DCRs, we discuss how different coding modes in motion compensation are modified in Section 2.4.1. Finally, we present experimental results and conclusions in section 2.5 and 2.6, respectively.

2.2 Related work

It was argued in [26] that since depth maps in texture-plus-depth multiview video are only used for view synthesis and not themselves directly viewed, synthesized-view-specific metrics should be used during depth map coding optimizations. [26] proposed alternative mode selection strategies in H.264 when coding depth maps, so that the distortion term reflects distortion in the synthesized view rather then distortion of the depth maps themselves.

Observing that depth maps are mostly flat surfaces with sharp edges, alternative coding schemes have also been proposed [34,50]. [34] proposed edge-adaptive wavelets, and [50] proposed edge-adaptive transforms, where the goal in both schemes is to avoid filtering across depth edges, which would result in many hard-to-code high frequency components in the transform domain. We differ from these works in that we focus on reducing the energy of the prediction residual during motion compensation, given that each depth pixel only needs to be reconstructed within a well defined tolerable range.

Don't care regions have been originally defined for finding the sparsest representation of transform coefficients in the spatial dimension in [14]. There, given per-pixel tolerable range for reconstruction (don't care regions) in a code block, the sparsest transform domain representation of depth signal is sought by minimizing the l_0-norm. In this work we extend the approach in [14] by exploiting the degrees of freedom defined in DCRs to seek coding gain in the *temporal* dimension for depth video. How to jointly optimize depth video in both spatial and temporal dimension given per-pixel DCR is left for future work.

2.3 Don't care region: definition

In the texture+depth video format, each camera-captured view $n = 1, \ldots, N$ is represented by one texture and one depth map. If the images are properly rectified (i.e., they are warped so that one captured image is a pure horizontal shift of another), then depth can be easily converted to disparity information, which is proportional to the inverse of depth. In the following, we will use both "disparity" and "depth" to refer to the disparity map at each view. Given $\mathbf{v}_n$, $\mathbf{v}_{n+1}$ and $\mathbf{d}_n$, $\mathbf{d}_{n+1}$, texture and disparity maps at views n and $n+1$, respectively, it is possible to synthesize any texture map $\mathbf{v}_k$ at intermediate view k, $k \in [n, n+1]$, using a depth-image-based rendering (DIBR)

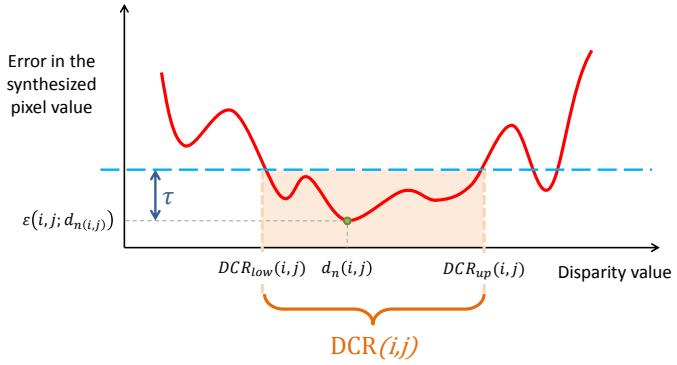


Figure 9: Definition of DCR for a given threshold τ.

algorithm such as [35]. Essentially, any DIBR algorithm synthesizes a pixel value in $\mathbf{v}_k$ by properly mapping corresponding pixels from texture maps $\mathbf{v}_n$ and $\mathbf{v}_{n+1}$, according to their disparity. If no corresponding pixels in $\mathbf{v}_n$ and $\mathbf{v}_{n+1}$ are found (due to dis-occlusion), then an inpainting technique can be used to fill in the missing pixels using neighboring pixel information. If the captured cameras are close to each other, however, then the number of dis-occluded pixels is expected to be small.

Since the disparity values are used as geometric information for pixel mapping during DIBR (and geometry of the captured scene varies greatly across space), not all the disparity pixels need be reconstructed with the same fidelity in order to guarantee a certain quality in the synthesized view. For example, depth pixels corresponding to smooth areas can be reconstructed with less accuracy than pixels at foreground object boundaries, as errors in the latter would produce large distortion when the decoder errs in mapping foreground textural pixels to background and vice versa. We formalize this concept as "don't care region" in the next paragraph.

We now define per-pixel DCRs for depth map $\mathbf{d}_n$, assuming target synthesized view is n. In other words, we consider the case $k = n$, since the encoder does not known which viewpoint k will be chosen at the decoder. This is generally the most difficult scenario, since the target view is the farthest from the reference. A similar procedure can be done for depth map $\mathbf{d}_{n+1}$ assuming target synthesized view $n + 1$.

A pixel $v_n(i, j)$ in texture map $\mathbf{v}_n$, with associated disparity value $d_n(i, j)$, can be mapped to a corresponding pixel in view $n + 1$ through a view synthesis function $s(i, j; d_n(i, j))$. In the simplest case where the views are captured by purely horizontally shifted cameras, $s(i, j; d_n(i, j))$ corresponds to a pixel in texture map $\mathbf{v}_{n+1}$ of view $n+1$ displaced in the x-direction by an amount proportional to $d_n(i, j)$; i.e.,

$$s(i, j; d_n(i, j)) = v_{n+1}(i, j - \gamma \cdot d_n(i, j)) \tag{1}$$

where γ is a scaling factor depending on the camera spacing.

We now define *view synthesis error*, $\varepsilon(i, j; d)$, as the absolute error between the mapped-to pixel $s(i, j; d)$ in the synthesized view $n + 1$ and the mapped-from pixel

$v_n(i,j)$ in v_n, given disparity value d for pixel (i,j) in v_n; i.e.,

$$\varepsilon(i,j;d) = |s(i,j;d) - v_n(i,j)|\,. \tag{2}$$

If $\mathbf{d}_n$ is compressed, the reconstructed value $\tilde{d}_n(i,j)$ employed for view synthesis may differ from $d_n(i,j)$ by an amount $e(i,j) = \tilde{d}_n(i,j) - d_n(i,j)$, resulting in a (generally larger) view synthesis error $\varepsilon(i,j;d_n(i,j)+e(i,j)) > \varepsilon(i,j;d_n(i,j))$. We define the *Don't Care Region* $\mathrm{DCR}(i,j) = [\mathrm{DCR}_{low}(i,j), \mathrm{DCR}_{up}(i,j)]$ as the *largest* contiguous interval of disparity values containing the ground-truth disparity $d_n(i,j)$, such that the view synthesis error for any point of the interval is smaller than $\varepsilon(i,j;d_n(i,j)) + \tau$, for a given threshold $\tau > 0$. The definition of DCR is illustrated in Figure 9. Note that DCR intervals are defined *per pixel*, thus giving precise information about how much error can be tolerated in the disparity maps. We also remark that the DCRs can be computed at the encoder side since both the views and the associated disparities are available.

2.4 Motion Prediction using DCR

The defined per-pixel DCRs give us a new degree of freedom in the encoding of disparity maps, where we are only required to reconstruct each depth pixel at the decoder to within its defined range of precision (as opposed to the original depth pixel), thus potentially resulting in further compression gain. Specifically, we change three aspects of the encoder in order to exploit DCRs: i) motion estimation, ii) residual coding, and iii) skip mode.

2.4.1 Motion estimation

During motion estimation for depth map encoding, the encoder searches, for each target block $\mathcal{B}$, a corresponding predictor block $\mathcal{P}$ in a reference frame which minimizes the Lagrangian cost function

$$\mathcal{P}^* = \arg\min_{\mathcal{P}} D_{\text{MV}}(\mathcal{B},\mathcal{P}) + \lambda_{\text{MV}} R_{\text{MV}}(\mathcal{B},\mathcal{P}), \tag{3}$$

where $R_{\text{MV}}(\mathcal{B},\mathcal{P})$ is the bit overhead required to code the motion vector from position of $\mathcal{P}$ to $\mathcal{B}$, and λ_{MV} is a Lagrange multiplier. The term $D_{\text{MV}}(\mathcal{B},\mathcal{P})$ is a measure of the energy of the prediction residual $r(i,j) = \mathcal{P}(i,j) - \mathcal{B}(i,j)$ for each pixel (i,j) in the target block $\mathcal{B}$ and the corresponding pixel in the predictor block $\mathcal{P}$. Typical choices for measuring the energy of residuals include the sum of absolute or squared differences — SAD or SSD, respectively.

For a given predictor block $\mathcal{P}$, we can reduce the energy of the prediction residuals using defined per-pixel DCRs as follows. We first define a per-block *DCR space* for a target block $\mathcal{B}$ as the feasible space containing depth signals with each pixel falling inside its per-pixel DCR. As an example, Figure 10 illustrates the DCR space for a two-pixel block with per-pixel DCR $[2,6]$ and $[1,4]$. For a given predictor block, to minimize the energy of the prediction residuals, we *identify a signal in DCR space closest to the predictor signal in Euclidean distance.* In Figure 10, if the predictor is

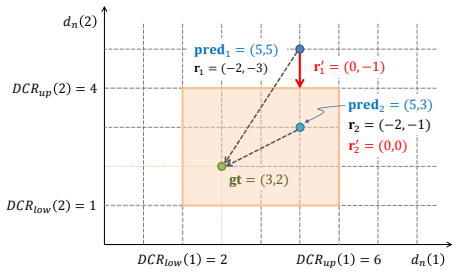


Figure 10: Coding the residuals using DCR with a toy example with just two pixels ($d_n(1)$ and $d_n(2)$). In conventional coding, given predictor (**pred**), one aims to reconstruct the original ground truth (**gt**). However, considering DCR, it is sufficient to encode a generally smaller residual, i.e. one that enables to reconstruct a value inside or on the border of the DCR (shaded area in the picture).

$(5, 5)$, we identify $(5, 4)$ in DCR space as the closest signal in DCR space, with resulting residuals $(0, -1)$. If the preditor is $(5, 3)$, we identify $(5, 3)$ in DCR space as the closest signal with residuals $(0, 0)$.

In mathematical terms, we compute a prediction residual $r'(i, j)$ for each pixel (i, j) given predictor pixel value $\mathcal{P}(i, j)$ and DCR $[\mathrm{DCR}_{low}(i, j), \mathrm{DCR}_{up}(i, j)]$ according to the following soft-thresholding function:

$$r'(i,j) = \begin{cases} \mathcal{P}(i,j) - \mathrm{DCR}_{up}(i,j) & \text{if } \mathcal{P}(i,j) > \mathrm{DCR}_{up}(i,j), \\ \mathcal{P}(i,j) - \mathrm{DCR}_{low}(i,j) & \text{if } \mathcal{P}(i,j) < \mathrm{DCR}_{low}(i,j), \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

We then use the residuals $r'(i, j)$ with respect to DCR to calculate D_{MV} in (3). If SAD is used as distortion metric, we get:

$$D'_{\mathrm{MV}} = \sum_{(i,j)\in\mathcal{B}} |r'(i,j))|. \tag{5}$$

Since the distortion D_{MV} is now zero for any motion vector which points to a predictor inside DCR, the encoder can select from a potentially larger set of zero-distortion candidate predictors. Among them, the one with the smallest rate term R_{MV} will result in a small Lagrangian cost.

2.4.2 Coding of prediction residuals

Once the optimal predictor $\mathcal{P}^*$ for a given target block has been found, we encode r' with respect to the per-block DCR, in place of the residuals r computed with respect

to ground truth depth signal. Notice that this applies also to INTRA coding modes as well. Although the prediction technique is different from the case of INTER modes (spatial prediction is used instead of temporal prediction), we still encode the residue that enables to reconstruct a value inside the DCR which is as close as possible to the predictor. Since this criterion is applied to any pixel in a block, we are in fact coding the residuals with respect to DCR having minimum energy (ℓ_2 norm). In general, since both rate and distortion terms are computed using minimum-energy residuals r' for inter and intra modes, the actual selected mode for a given target block will be different from the one selected when coding residuals with respect to the ground truth signal.

We note that, although computing minimum-energy prediction residuals from (4) is computationally convenient (in fact, its complexity grows linearly with the number of pixels), this is not necessarily the best possible strategy in terms of rate-distortion performance. This is because the minimum-energy residual may not lead to the lowest transform coding rate; e.g., lower-energy residuals $\{1, 0, 1, 1\}$ leads to coding of more non-zero transform coefficients (thus higher rate) than higher-energy residuals $\{1, 1, 1, 1\}$. Therefore, the best motion vector and the best coding residuals for a given block with defined per-block DCR is a joint optimization problem, whose solution is not trivial. We leave the investigation of this problem for future work.

2.4.3 Skip mode

The coding of prediction residuals for INTER/INTRA modes described in the previous section guarantees that the reconstructed block will be within DCR (up to quantization errors). If the SKIP mode is selected instead, the prediction residuals are not coded. Thus, the reconstructed pixels could be potentially far away from DCR. This is potentially harmful since, by construction, there is no upper bound to the distortion in the synthesized view when a depth pixel is reconstructed outside DCR. This requires SKIP mode to be handled differently from INTER/INTRA.

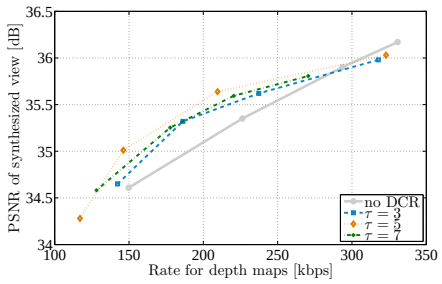
In order to be sure that distortion in the synthesized view will be bounded in SKIP macroblocks, we prevent the SKIP mode to be selected from the encoder if *any* reconstructed pixel of that macroblock violates DCR. More formally, we alter the distortion term D_{MD} in the Lagrangian function used for mode decision according to the following barrier penalty function:

$$D'_{\text{MD}} = \begin{cases} 0 & \text{if } r'(i,j) = 0 \quad \forall i,j; \\ +\infty & \text{otherwise.} \end{cases} \tag{6}$$

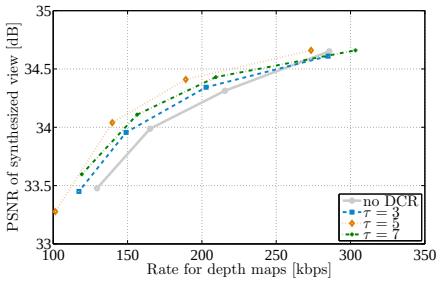
Although this could be conservative in terms of rate optimization, it guarantees that the distortion in the synthesized view for SKIP macroblocks will be bounded by τ.

2.5 Experimental results

We modified an H.264/AVC encoder (JM reference software v. 18.0) in order to include DCR in the motion prediction and coding of residuals. Our test material includes 100



(a) *Kendo*



(b) *Balloons*

Figure 11: RD curves for *Kendo* and *Balloons*

frames of two multiview video sequences, *Kendo* and *Balloons*,[2] with spatial resolution of 1024×768 pixels and frame rate equal to 30 Hz. For both sequences we coded the disparity maps $\mathbf{d}_3$ and $\mathbf{d}_5$ of views 3 and 5 (with IPP...GOP structure), using either the original H.264/AVC encoder or the modified one. In the latter case, we computed per-pixel DCRs with three values of τ, namely $\tau = \{3, 5, 7\}$. Given the reconstructed disparities in both cases (with/without DCR), we synthesize view $\mathbf{v}_4$ using the uncompressed views $\mathbf{v}_3$ and $\mathbf{v}_5$ and the compressed depths $\tilde{\mathbf{d}}_3$ and $\tilde{\mathbf{d}}_5$. Finally, we evaluate the quality of the reconstructed view $\hat{\mathbf{v}}_4$ w.r.t. ground-truth center view $\mathbf{v}_4$.

The resulting rate-distortion curves are reported in Figure 11. For the *Kendo* sequence, using $\tau = 5$ we obtain an average gain in PSNR of 0.34 dB and an average rate saving of about 28.5%, measured through the Bjontegaard metric. Notice that the proposed method enables a significant amount of bit saving by reducing *selectively* the fidelity of the reconstructed depths where this is not bound to affect excessively the synthesized view. On the other had, to achieve an equivalent bitrate reduction, a conventional decoder should quantize prediction residuals much more aggressively, and the quantization error can affect *all* the synthesized pixels.

In order to show the impact of the proposed method on the choice of motion vectors

[2] Available at http://www.tanimoto.nuee.nagoya-u.ac.jp/

Table 2: Coding statistics for two RD points of *Kendo*

	bitrate [kbps]	PSNR [dB]	% SKIP	Motion info. [bit/frame]	Residuals [bit/frame]
no DCR	230.4	33.99	80.20	582.10	522.41
DCR ($\tau = 5$)	179.5	34.04	92.25	253.48	240.62

and optimal modes at the encoder, we show in Table 2 the coding statistics of two RD points in Figure 11(a). We notice that most of the rate savings are obtained through a more efficient use of SKIP mode (which increases by over 18% in this case), and by a more efficient prediction of motion and coding of residuals. Observe that in the current setting, we are not taking into account the effect of quantization error, which could make reconstructed values lie outside DCR. We will investigate how to push the de-quantized and reconstructed values inside DCR in future work.

2.6 Conclusions

Depth maps need not be reproduced with high fidelity at the decoder in order to synthesize novel views with acceptable quality. In this contribution we have formalized this intuition by defining per-pixel don't care regions. DCRs provide new degrees of freedom to the encoder, which can result in a higher coding efficiency of depth. Specifically, we demonstrated that DCR-aware motion compensation and coding of residuals can lead to substantial coding gains with respect to state-of-the-art video coding paradigms.

In fact, motion compensation and coding of residuals is a joint estimation problem, since any value inside DCR is a feasible reconstruction point which could entail a different RD cost. Also, quantization may move reconstructed values outside DCR, causing a deterioration of the synthesized video's quality. Solving both these two issues is the focus of our current research.

3 Improving 3D-HEVC via the modification of the Merge candidate list

The emrging HEVC-based 3D compression standard (3D-HEVC) [57] exploits spatial, temporal, inter-component and inter-view redundancies to efficiently encode the 3D video. Inter-view redundancies are in particular exploited by disparity-compensated prediction (DCP), currently present in the MVC standard [12]. DCP enables having, for the currently frame, reference frames from different views at the same time instant. The vector of a prediction unit (PU) pointing to a PU in a different view is called a disparity vector (DV), while a vector pointing to a reference frame in the same view but at a different time instant is called a motion vector (MV). 3D-HEVC uses the Merge coding mode [24] introduced in HEVC which establishes a list of candidate vector predictors to efficiently reduce the signaling cost of motion / disparity parameters (vectors + reference indices). This list rarely contains DV predictors, and although there is a multiview candidate in the list, it is preferred to be a MV than a DV. This MV / DV asymetry highly penalizes DCP, which remains largely less selected than motion-compensated prediction. In this contribution, we modify the Merge candidate list by inserting a new candidate which is always a DV. The DV candidate is inserted either in the secondary (method 1) or the primary (method 2) list of Merge candidates.

3.1 Background

The Merge coding mode in 3D-HEVC allows a PU to inherit the motion / disparity parameters from a neighboring PU. Motion / disparity parameters from different neighboring PUs form the Merge candidate list. Only the index of the most coding efficient candidate is sent in the bitstream, along with an optional PU residual. Merge mode thus creates contiguous motion / disparity areas at a minimal cost in 4 different dimensions: horizontal, vertical, temporal, and inter-view.

3D-HEVC uses the Merge candidate list of HEVC [8], which consists, in order, of four spatial candidates and one temporal candidate. A pruning process is performed within the spatial candidates to remove redundant vectors [6]. 3D-HEVC adds a so-called multiview candidate, only for dependent views, in the first position of the list. If some of these 6 candidates are unavailable (the PU corresponding to the position falls outside the slice, or is Intra-coded, or the candidate is redundant), a secondary list of candidates is computed. These candidates are then appended to the list so that the total number of candidates is always 6. The candidates in that secondary list are, in order, combined candidates from mixed primary vectors of both reference lists, and zero-vector candidates, each having a different reference index.

The multiview candidate is computed in the following manner: first, a DV is derived for the current PU. In previous versions of 3D-HEVC, a depth map estimate was maintained for each view and the DV was derived from the highest depth value contained in the co-located PU in the estimate. Currently in 3D-HEVC, to reduce complexity, a simple neighbor search for a DV is performed. The DV allows finding a reference PU in the main view that corresponds to the current PU in the side view. The motion

vector of that reference PU is then set as the multiview candidate. If the reference PU is intra-coded or falls outside the slice, the DV itself is set as the multiview candidate. Thus, there is always a temporal preference for this candidate, and consequently, the Merge list is in most cases composed only of MVs.

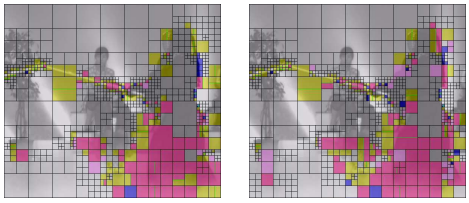
Several tools that modify the Merge candidate list construction in 3D-HEVC were proposed to either achieve coding gains, or reduce complexity / memory consumption: In [23], the primary candidate list is checked and the first DV candidate found is used to compute, by adding a positive and a negative offset, two more DV candidates which will then be added to the list. However this requires having a DV in the primary list to begin with, which is not a frequent case. Consequently, the coding gains are limited. The Merge pruning process can also be changed, like in [29], where a comparison between the inter-view candidate and the first two spatial candidates is added. This method achieved 0.3% bitrate reduction on dependent views with no runtime increase and was adopted in 3D-HEVC. For depth PU coding, the first Merge candidate was modified in [62] to refer to merging with the co-located texture PU, as the texture and depth motion information are highly correlated. A 1.1% bitrate reduction was reported for coded and synthesized views. Hence, this Motion Vector Inheritance tool was adopted in 3D-HEVC.

Tools that affect the Merge candidate list construction were also proposed in HEVC. In [30], the temporal candidate (TMVP) position is changed from the center of the co-located PU to the bottom-right position. Significant bitrate reductions of 0.9% were reported and thus the method was adopted in HEVC. In [63], two refined candidates were computed from the first Merge candidate and added to the secondary list of candidates, to replace the combined ones for uni-predicted PUs. Coding gains were not significant enough however to favor adoption.

These methods try to improve the candidate list construction but with no particular intention to balance the DCP selection against the MCP selection in the process. We propose, as described in the next section, a novel method to reach a better DCP / MCP equilibrium by inserting a DV candidate in the Merge list.

3.2 Adding a disparity vector to the Merge list

Our idea is based on the observation that DCP is not often selected by the HTM encoder. Also, Merge mode was observed to be often selected for coding PUs. This can be seen in Figure 12(a) which shows parts of a B-frame of the Kendo sequence coded with the reference HTM encoder. The PUs coded using MCP are shown in grey (Merge-SKIP) and green (Inter). PUs coded using DCP are shown in light pink (Merge-SKIP) and dark pink (Inter). Blue PUs are coded in Intra. We can clearly see that Merge mode is selected often, and that DCP coded PUs are not numerous. Table 3 gives the percentages of Merge coded PUs, DCP coded PUs, and DCP coded PUs in Merge mode, in dependent texture and depth views, averaged across four QPs, of seven MPEG sequences. These results confirm the assertion that Merge mode is selected often, actually for 92% of PUs. This is due to the fact that Merge mode is very efficient at reducing the cost of motion / disparity parameters as only an index is encoded. Table 3 also shows that only 16% of PUs use DCP, and they are also most



(a) Reference (b) Proposed method

Figure 12: CU coding modes in parts of a Kendo B-frame with a reference coding and with our proposed method

Sequence	**Merge**	**DCP**	**DCP-Merge**
Kendo	92	17	14
Newspaper	88	15	12
Balloons	93	13	11
Dancer	90	26	21
GT Fly	96	18	15
Poznan Hall2	95	9	7
Poznan Street	94	15	12
Average	**92**	**16**	**13**

Table 3: Percentage of Merge coded PUs, DCP coded PUs, and DCP coded PUs in Merge mode

often coded in Merge mode (13%). While it is true that there are often more temporal correlations than inter-view, as shown in [68], the main issue behind the unfrequent DCP selection remains the lack of accurate DV predictors. Indeed, DCP can yield a better prediction for a given PU than MCP, in case there is little disparity between views or if there is fast motion in the video. However, not having a DV predictor in the Merge list increases the rate needed to code the PU with DCP since the only option left is to send a motion vector residual. MCP, while maybe not yielding a lower distortion value, requires a lower rate due to the fact that there are numerous MV predictors in the Merge list and signaling the motion parameters only costs an index. Consequently MCP is chosen more often since its Lagrangian cost is smaller, but if a DV predictor was added in the Merge list, as proposed in this contribution, the required rate for DCP coding would be decreased, hence increasing the selection of DCP and achieving coding gains.

When computing the multiview candidate in the Merge list, a DV pointing to a reference block in the base view is derived, as explained in Section 3.1. The multiview candidate is set as the MV of that reference block, and if that MV does not exist, it is set as the DV. We propose to insert that DV as a new interview candidate in the Merge list along side the multiview candidate if the latter turned out to be a MV.

Two insertion methods are proposed. In method 1, the candidate is inserted in

the secondary list along with the combined and the zero vector candidates. If any of the first five candidates in the primary list is unavailable, the interview candidate is inserted after the final spatial candidate (before the temporal) to complete the list. If more primary candidates are unavailable, the combined and zero vector candidates are then appended to the list, as it is normally done. In method 2, the candidate is inserted in the primary list, in the 5^{th} position, shifting the final spatial candidate to the 6^{th} position. The temporal candidate is hence pushed out of the primary list and into the secondary list. It is the first candidate in the secondary list to be appended back in the primary list if some candidates are unavailable. Figure 13 illustrates these two methods. In both methods, before inserting the interview candidate, a redundancy check with all candidates preceding it in the list is performed for better coding efficiency. Note that the insertion positions in both methods have been empirically set as those positions gave out the most coding gains on average.

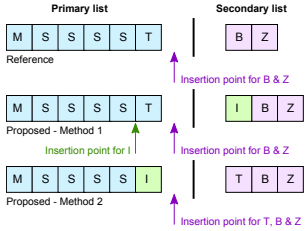


Figure 13: Proposed insertion methods (M: multiview, S: spatial, T: temporal, B: combined, Z: zero, I: interview candidate)

3.3 Experimental results

We have implemented our two proposed methods in HTM-4.1 [56]. We have strictly followed the common test conditions (CTCs) defined by JCT-3V [45]. A GOP of 8 was considered with an Intra period of 24. Four QP combinations for texture and depth (respectively) were considered: (25;34), (30;39), (35;42) and (40;45) to conform to CTCs. We have tested the two methods on seven sequences defined in the CTCs (1920×1088 and 1024×768). Experiments were done on 10 seconds of video length. Each sequence is composed of 3 texture and 3 depth views (one central base view and two side views). After encoding, 3 intermediate views were synthesized between the left and the center view, and another 3 between the center and the right views. PSNR on synthesized views were computed with respect to synthesized views rendered with uncompressed original texture and depth views. Coding gains are measured with the Bjontegaard delta (BD-Rate) metric [7].

Tables 4 and 5 give the coding gains (negative values are gains) and runtimes obtained with methods 1 and 2 respectively. These results are summarized in Table 6 which also gives the average results if the redundancy check preceding the insertion of the interview candidate in the list is removed. In these tables, the "Video" column shows the gains on the central (0) and on the two side views (1 and 2) and averages

these results. The "Synt." column gives results on the 6 synthesized views (the bitrate considered is the sum of the 3 texture and depth bitrates, and the PSNR is the average PSNR of all 6 synthesized views). The "Coded+Synt." result is the same as in the previous column except that the PSNR considered is the average PSNR of the 6 synthesized views and the 3 coded texture views.

Sequence	**Video**				**Synt.**	**Coded**	**Runtimes**	
	0	**1**	**2**	**Avg**		**+Synt**	**Enc**	**Dec**
Balloons	0.0	-0.6	-0.6	-0.3	-0.2	-0.2	96	102
Kendo	0.0	-0.6	-0.4	-0.2	-0.1	-0.2	100	101
Newspaper	0.0	-0.3	-0.3	-0.1	-0.1	-0.1	100	101
GT Fly	0.0	-1.2	-1.0	-0.3	-0.2	-0.3	97	100
Poznan Hall2	0.0	0.2	-0.5	-0.1	-0.1	-0.1	97	94
Poznan Street	0.0	-0.6	-0.6	-0.2	-0.2	-0.2	90	100
Dancer	0.0	-0.6	-0.6	-0.2	-0.2	-0.2	97	102
Average	**0.0**	**-0.5**	**-0.6**	**-0.2**	**-0.2**	**-0.2**	**97**	**100**

Table 4: Bitrate reduction per sequence, in %, with method 1

Sequence	**Video**				**Synt.**	**Coded**	**Runtimes**	
	0	**1**	**2**	**Avg**		**+Synt**	**Enc**	**Dec**
Balloons	0.0	-0.6	-0.6	-0.3	-0.2	-0.2	97	95
Kendo	0.0	-0.6	-0.4	-0.2	-0.1	-0.1	98	101
Newspaper	0.0	-0.3	-0.2	-0.1	-0.1	-0.1	100	90
GT Fly	0.0	-1.2	-1.2	-0.4	-0.2	-0.3	101	100
Poznan Hall2	0.0	-0.1	-0.3	-0.1	-0.1	-0.1	93	107
Poznan Street	0.0	-0.7	-0.6	-0.2	-0.2	-0.2	92	100
Dancer	0.0	-0.6	-0.6	-0.2	-0.2	-0.2	93	101
Average	**0.0**	**-0.6**	**-0.6**	**-0.2**	**-0.2**	**-0.2**	**96**	**99**

Table 5: Bitrate reduction per sequence, in %, with method 2

Tables 4 and 5 show bitrate reductions of 0.5% (resp. 0.6%) and 0.6% for side views, 0.2% for synthesized and 0.2% for coded and synthesized views. This is accompanied by a 3% (resp. 4%) encoder runtime reduction. No gains are achieved on the central view since our method is not applied there. Table 6 shows that coding efficiency is reduced if the redundancy check is removed, with no decrease in encoder and decoder runtimes compared to the original version.

The gains obtained result from an increase in DCP selection. Inserting a DV into the Merge candidate list reduces the rate needed for DCP coding and favors its selection, especially if there is small disparity between views (interview redundancies are much higher, and DVs can point to a better hypothesis) or if there is fast motion in the video (MVs are not able to correctly predict PUs). Figure 12(b) indeed shows an increase in

Method	Video				Synt.	Coded +Synt	Runtimes	
	0	1	2	Avg			Enc	Dec
M1	0.0	-0.5	-0.6	-0.2	-0.2	-0.2	97	100
M1-NO RC	0.0	-0.4	-0.4	-0.1	-0.1	-0.1	98	101
M2	0.0	-0.6	-0.6	-0.2	-0.2	-0.2	96	99
M2-NO RC	0.0	-0.5	-0.4	-0.2	-0.1	-0.1	98	100

Table 6: Bitrate reductions when the redundancy check is removed (NO RC) in method 1 (M1) and 2 (M2)

DCP coded PUs compared to Figure 12(a). This is confirmed in the numerical results of Table 7, which shows, for both methods, an increase of 8% and 11% on average in the percentage of DCP-coded PUs and DCP-coded PUs using Merge mode.

Sequence	DCP increase		DCP-Merge increase	
	M1	M2	M1	M2
Kendo	6.2	6.4	9.1	8.9
Newspaper	5.0	4.9	7.8	8.0
Balloons	8.2	7.8	12.2	11.3
Dancer	7.7	7.3	10.8	10.6
GT Fly	17.6	17.0	21.0	20.3
Poznan Hall2	6.3	6.8	8.4	8.9
Poznan Street	6.9	8.2	9.3	10.8
Average	**8.3**	**8.3**	**11.2**	**11.3**

Table 7: Percentage increase of DCP-coded PUs and DCP-coded PUs using Merge mode in the two methods

The complexity resulting from the redundancy check used in the two methods is debatable. The purpose of this redundancy check is to avoid having a redundant DV candidate in the list which will either push potentially better primary candidates further down the list, while increasing their indices, and hence their signaling cost, in the process, or take the place of other, potentially better, secondary candidates which will not even be evaluated. The maximum number of checks equals 4 and 5 in method 1 and 2 respectively. These would be quite complex to perform for each PU. However, we show in Table 6 that removing the redundancy check decreases coding efficiency, as expected, while not reducing neither encoder or decoder runtime. Indeed, the worst case rarely occurs. Consequently, keeping the redundancy check is a better choice.

The two methods also brought small encoder runtime reductions of 3 and 4%. This is because inserting a DV candidate in the Merge list means constructing one less secondary candidate, which is a complex process since it involves mixing different vectors to construct combined candidates or looping around all reference indices to construct zero-vector candidates. Additional experiments have shown that the number

of constructed secondary candidates has decreased by 9% in the two methods.

3.4 Conclusion

In this part, we have presented a novel method to improve the selection of DCP for dependent views in 3D-HEVC. A DV candidate has been inserted in the Merge candidate list in order to reduce the rate required for DCP, hence favoring its selection. Two insertion methods have been proposed, one where the DV is inserted in the secondary candidate list and another where the DV is inserted in the primary list. Bitrate reductions of 0.5% (resp. 0.6%) and 0.6% for the two side views, along with 0.2% for synthesized and for coded+synthesized views are reported. These were accompanied by a 3% (resp. 4%) encoder runtime reduction since secondary candidates are less required to be constructed. Both methods were presented at the 2nd JCT-3V meeting and method 2 was adopted in 3D-HEVC.

The gains obtained highly depend on the quality of the derived DV. The DV derivation process is the same as the one used for the multiview candidate. Improving this process can lead to coding gains due to the improvement of the multiview candidate which depends on it, but also due to the improvement of our newly added interview candidate.

4 Modification of the disparity vector derivation in 3D-HEVC

As shown in the previous section, 3D-HEVC exploits spatial, temporal, inter-component and inter-view dependencies in order to efficiently code the 3D information. In particular, inter-view redundancies between a currently coded dependent view and a previously coded base view which serves as a reference, are exploited using the Inter-View Motion Prediction (IVMP) and the Inter-View Residual Prediction (IVRP) coding tools [57]. In both methods, a currently coded prediction unit (PU) in a dependent view is compensated, in the view axis, using a disparity vector (DV) in order to find its corresponding PU in the base view.

The Merge coding mode [24], initially introduced in HEVC, allows a PU to inherit the motion parameters (motion vectors + reference indices) of a neighboring PU. A candidate list, composed of the motion parameters of four spatial neighbors and one temporal neighbor, is formed and and the index of the most coding efficient candidate in the list is sent in the bitstream with an optional PU residual.

In 3D-HEVC, IVMP expands the Merge candidate list in the view axis by adding a multiview neighbor. Indeed, after finding the base PU using a derived DV, its motion vector, if it exists, is inserted in the first position of the Merge candidate list of the current PU. This candidate is commonly referred to as the multi-view candidate. The DV used to find the base PU is also inserted as an inter-view candidate in the fourth position [40] of the list. If the base MV does not exist (the base PU is coded in Intra mode for instance), the DV is set as the multi-view candidate and the inter-view candidate does not exist. The base MV and the DV are also inserted in the AMVP candidate list [24] where they are used as predictors for, respectively, the MV or DV of the current PU. A vector residual is thus transmitted in this case. In IVRP, the residual samples of the base PU are used to predict the residual samples of the current PU in order to further reduce the residual energy for more efficient compression.

The DV used for disparity compensation in IVMP and IVRP can be estimated. Multiple disparity estimation techniques ranging from classic block matching algorithms to more advanced stereo matching methods [48] or convex optimization approaches under illumination variations [39] can be used. However estimating the DV necessarily implies transmitting it in the bitstream for decodability. To avoid this costly transmission, the DV can be derived using already coded information. Specifically, in the current 3D-HEVC draft, it is derived using a search process for a DV across spatio-temporal neighboring positions. This neighboring position setup has been used until now in video coders for deriving a MV predictor [28], or a MV for direct inheritance [24].

The first DV found in this process, called Neighbor Disparity Vector (NBDV), is selected with no guarantee of optimality. Various methods have been proposed to improve NBDV, but none dealt with the sub-optimality induced by selecting the first DV found in the search process. In this contribution, we propose a solution to this problem with a method that first stores the DVs of all the checked neighbors in a single list. Second, the redundant vectors are removed from this list, and finally, the median

of the remaining vectors is computed and is set as the final DV which will be used for IVMP (for IVRP, the method is not applied, the first found DV is selected).

The rest of this contribution is organized as follows: Section 4.1 presents the state of the art in DV derivation processes for 3D-HEVC. Section 4.2 describes the proposed method and its variants. The corresponding results are presented in Section 4.3 with a detailed interpretation. Section 4.4 concludes this part while underlining possibilities for future work.

4.1 State of the art

In this section, we will detail the different DV derivation processes used in 3D-HEVC. We will present in particular the currently used NBDV method which we improve in this work.

In 3D-HEVC, the texture component is coded before the depth component. Consequently, when coding a PU in a dependent view, the DV pointing to the corresponding PU in the base view cannot be computed from the depth component because it has not been coded yet. Getting the DV from the original depth map would require transmitting it in the bitstream because otherwise, the process cannot be repeated at the decoder. In order to avoid this costly transmission, a depth map estimate is computed and maintained for each view using already coded texture information. The maximum depth value contained in the collocated PU in the depth map estimate is transformed into the required DV. This derivation process is called Depth Map Disparity Vector (DMDV) [57]. To obtain the depth map estimates, the coded disparity vector field between the first dependent view and the base view is transformed into a depth map which is then warped to the base view and to other dependent views. Over time, the estimated depth maps are motion-compensated using the same motion vector field as in texture and corrected with coded disparity vector fields.

The complex warpings and successive motion compensations that DMDV involves led to the development of a lighter, less complex derivation process: NBDV [66]. NBDV is a simple search process across neighboring positions. The PUs covered by these positions are checked if they are coded using disparity-compensated prediction (DCP), in which case they have a DV, and the first DV found is selected as the final DV used for IVRP and IVMP. The positions are depicted in Figure 14. There are 5 spatial

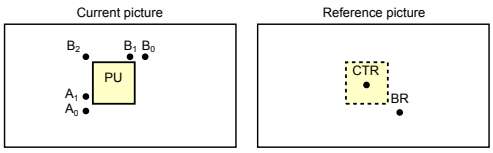


Figure 14: Spatial and temporal neighboring positions in NBDV

positions denoted by A_1 (left), B_1 (above), B_0 (above-right), A_0 (below-left)and B_2 (above-left), checked in this order. A PU covered by one of these positions can have up to two vectors, one from each reference list (list 0 and list 1) and they are both checked

if they are DVs. Temporal positions are checked next, and they consist of the center of the collocated PU (CTR) and collocated bottom-right PU (BR) in a maximum of 2 temporal reference pictures. Furthermore, if a neighboring spatial PU was coded in motion-compensated prediction (*i.e*, the PU has a MV in a specific reference list, not a DV), its MV could have been computed using IVMP which necessarily involves the derivation of a DV. Indeed, the MV could have been inherited from the multi-view candidate in Merge mode or predicted using the multi-view MV predictor in AMVP. Constructing the multi-view candidate in both lists (Merge & AMVP) requires a prior derivation of a DV, which would then be linked to the current MV. These special DVs, called DDVs [55], are also checked after the temporal neighbors in the following order A_0, A_1, B_0, B_1, B_2. Compared to DMDV, NBDV brings small losses (0.1% on coded+synthesized views) while reducing encoder and decoder runtimes by 8% [67].

Depth-oriented NBDV (DoNBDV) [11] is an interesting refinement of the classic NBDV process. It uses the coded depth map of the base view to refine the DV obtained after the standard NBDV process. Basicaly, the DV obtained is used to point to the corresponding PU in the base depth view. The maximum depth value inside that PU is converted into another DV which will be used for IVRP and IVMP. DoNBDV achieves significant bitrate reductions compared to NBDV (0.4% on coded views and 0.3% on coded+synthesized views) but adds a non-negligible decoding dependency between the base depth view and the dependent texture view (indeed, if the base depth view is corrupted, the dependent texture view cannot be decoded).

A final DV derivation process can be conceived if the depth is coded before the texture component. This is possible using the flexible coding order (FCO) tool which allows to change the coding order in 3D-HEVC. In this case, the DV can simply be computed from the coded depth component (taking the maximum depth value in the collocated depth PU and transforming it into a disparity) without the need of transmitting it since the process can be repeated at the decoder.

The DV derivation process in 3D-HEVC is subject to intensive research and is expected to change over the course of the standardization phase. Following the 2nd JCT-3V meeting, the derivation process currently used in 3D-HEVC is NBDV since it is coding efficient, not complex, and does not introduce new decoding dependencies. However, NBDV is sub-optimal. Indeed, the first DV / DDV found in a neighboring PU during the NBDV search process is selected as the final DV and the search process stops. The remaining neighbors are not checked even if some have a DV / DDV which is better, rate-distortion (R-D) wise, than the selected one hence the sub-optimality of the process. The proposed method answers and solves this specific issue.

4.2 The disparity vector derivation process

4.2.1 Preliminary study

Table 8 shows the percentage of PUs coded in Merge mode using either the multi-view or the inter-view candidate in version 5.0.1 of the 3D-HEVC reference software: HTM, averaged across four QPs, for various tested sequences. The test conditions used are the same as the ones used to evaluate our method, which are described in Section 4.3.1.

We can see that the multi-view Merge candidate is largely selected in HTM-5.0.1 (for 57% of PUs coded in Merge mode, on average) since it is inserted at the first position in the list (the rate needed to code a merge index of 0 is small, hence the R-D cost of this candidate is small as well). The inter-view candidate is inserted further down the list and is thus selected less often (only 1%).

Sequence	**Multi-view**	**Inter-view**
Kendo	51.6	2.1
Newspaper	53.4	1.4
Balloons	59.9	1.6
Dancer	45.9	1.7
GT Fly	65.6	0.9
Poznan Hall2	61.5	0.9
Poznan Street	60.6	1.2
Average	**56.9**	**1.4**

Table 8: Percentage of PUs coded in Merge mode using the multi-view or the inter-view candidates

The efficiency of the multi-view and the inter-view candidate directly depends on the derived DV. If the DV quality is improved, the distortion associated to these two candidates will decrease, along with their R-D cost, hence increasing their selection and achieving coding gains. These gains will be significant since on the one hand, we are in general improving the Merge coding mode which is already widely selected (our experiments have shown that it is selected for 92% of PUs in the same test conditions), and on the other hand, we are improving the first candidate in the Merge list which is also largely selected as shown in Table 8. It is important to note that some of our gains will also come from improving these candidates in the AMVP list but those gains are small compared to the ones resulting from the improvement in the Merge list.

4.2.2 Proposed method description

In our method, the search process is never stopped. All the spatial and temporal neighbors are checked, in the usual order, and all found DVs and DDVs are stored together in a single list. A redundancy check is applied to remove redundant vectors in this list. Then, the median of all remaining vectors is computed and is set as the final DV used for IVMP. Applying the method for IVRP as well will be tested seperately in a variant. Figure 15 illustrates the different steps of our algorithm.

The advantage of our method is that it groups different types of DVs, namely DVs obtained from DCP-coded PUs and DDVs obtained from MCP-coded PUs, in a single list. This heterogeneity in lists is usually coding efficient. For instance, in the Merge candidate list, secondary candidates are constructed to fill the list if primary candidates are unavailable. Hence, two types of candidates can potentially be in the same list, namely primary and secondary candidates. This configuration has been proven to

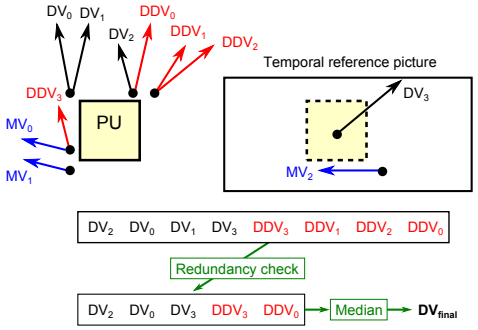


Figure 15: Proposed DV derivation method

bring coding gains compared to one which does not involve secondary candidates. Our method is thus set in the same mind frame.

The disadvantage of our method lies in the worst case scenario for median computation. Indeed, each spatial neighbor can have up to two vectors, one from each reference list, and there are 5 neighbors. In addition, there are two temporal neighbors in two temporal reference frames, each one having at most one vector. In case all spatial neighbors in both reference lists and all temporal neighbors have DVs or DDVs, and there is no redundancy between these vectors, the median has thus to be computed on 14 vectors. This is quite complex to perform in hardware. Consequently, in order to avoid this worst-case scenario, different variants of the method have been implemented and tested.

4.2.3 Variants

The 1REF variant consists of storing in the list a maximum of one vector per spatial neighbor. In case the spatial neighbor has two DVs or two DDVs, only the one from reference list 0 is stored. In case it has one DV and one DDV, only the DV is stored. In this configuration, the maximum number of spatial candidates is 5, making the worst-case maximum number of vectors on which the median is computed (*MaxCand*) equal to 9. Another variant, RMPOS, consists in simply removing one or more spatial positions (for example A_0) from the check, hence decreasing *MaxCand* by 2 (or by 1 if associated with 1REF) for each spatial position removed. In our experiments, RMPOS consisted in removing the A_0 and the B_2 spatial positions from the check. A final variant aimed at reducing *MaxCand*, called LIMIT-X, consists in storing only the first X found DVs / DDVs in the list. In this case, *MaxCand* = X. Note that the LIMIT-1 variant is equivalent to the standard NBDV process.

The following variants are not aimed at reducing *MaxCand*, but rather implemented and tested to make interesting interpretations: NODDV does not store any DDVs in the list, ALLOWRED removes the redundancy check before median computation, NOAMVP does not apply our method for AMVP while APPLYRES applies it for IVRP as well, and finally, MEAN replaces the median computation with the compu-

tation of the vectors' average.

4.3 Experimental results

4.3.1 Experimental setting

We have implemented the proposed method and all of its variants in HTM-5.0.1 [58]. We have strictly followed all the common test conditions (CTCs) defined by JCT-3V [46]. A GOP of 8 was considered with an Intra period of 24. Four QP combinations for texture and depth (respectively) were considered: (25;34), (30;39), (35;42) and (40;45) to conform to CTCs. We have tested the method and the variants on seven sequences defined in the CTCs (1920×1088 and 1024×768). Experiments were done on 10 seconds of video length. Each sequence is composed of three texture and three depth views (one central base view and two side views). After encoding, three intermediate views are synthesized between the left and the center view, and another three between the center and the right views. The renderer used is the one included in the HTM-5.0.1 package. This renderer, called "VSRS-1D-Fast", interpolates an intermediate view from a left and right reference. Remaining holes due to disocclusions are filled using a line-wise inpainting. PSNRs on synthesized views are computed with respect to synthesized views rendered with uncompressed original texture and depth views. Coding gains are measured with the Bjontegaard delta (BD-Rate) metric [7].

4.3.2 Coding gains

Objective results — Table 9 gives the coding gains (negative values are gains) achieved with our method. The anchor considered is HTM-5.0.1 under the same common test conditions. The results of our method are summarized also in Table 10 along side all the studied variants (only the average results accross all sequences are given in Table 10). In these tables, the "Video" column shows the gains on the central (0) and on the two side views (1 and 2) and averages these results. The "Synt." column gives results on the six synthesized views (the bitrate considered is the sum of the three texture and depth bitrates, and the PSNR is the average PSNR of all six synthesized views). The "Coded+Synt." result is the same as in the previous column except that the PSNR considered is the average PSNR of the six synthesized views and the three coded texture views. In Table 10, an additional column, "MaxCand" was added to show the maximum number of vectors on which the median can be computed in a worst-case scenario, per variant. Note that there is a ±3% error margin on the encoder and decoder runtimes, because even if launched back to back on the same machine, the runtime of an encoding or decoding process varies only slightly each time.

Table 9 shows 0.6% and 0.8% average bitrate reductions on the dependent views, and 0.2% on synthesized views, with a *MaxCand* of 14, as explained in Section 4.2.2. These gains were achieved with no increase on encoder or decoder runtimes. Note that no gains are reported on the central view because our method is simply not applied there (no DV derivation is done on the base view).

Table 10 shows the average coding results of three variants (1REF, 1REF+RMPOS, LIMIT-4) aimed at reducing *MaxCand*. These variants slightly reduce the gains ob-

Sequence	Video				Synt.	Coded +Synt	Runtimes	
	0	1	2	Avg			Enc	Dec
Balloons	0.0	-0.7	-0.7	-0.3	-0.2	-0.2	100	99
Kendo	0.0	-1.2	-1.3	-0.5	-0.4	-0.4	98	97
Newspaper	0.0	-0.7	-0.7	-0.3	-0.2	-0.2	99	98
GT Fly	0.0	-0.6	-0.9	-0.2	-0.2	-0.2	89	98
Poznan Hall2	0.0	0.0	-0.5	-0.1	-0.2	-0.2	102	101
Poznan Street	0.0	-0.4	-0.4	-0.1	-0.1	-0.1	104	95
Dancer	0.0	-0.9	-1.0	-0.3	-0.4	-0.4	108	99
Average	**0.0**	**-0.6**	**-0.8**	**-0.3**	**-0.2**	**-0.2**	**100**	**98**

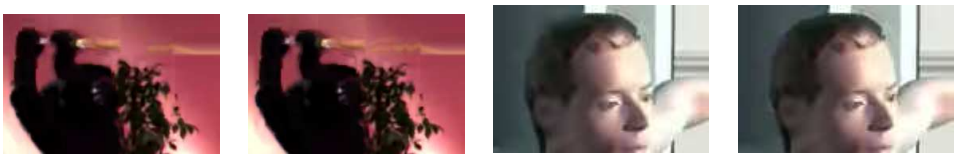
Table 9: BD-Rate coding results per sequence, in %, with the proposed method

Variant	Max Cand	Video				Synt.	Coded +Synt	Runtimes	
		0	1	2	Avg			Enc	Dec
Method	14	0.0	-0.6	-0.8	-0.3	-0.2	-0.2	100	99
1REF	9	0.0	-0.6	-0.7	-0.2	-0.2	-0.2	97	98
1REF+RMPOS	7	0.0	-0.5	-0.6	-0.2	-0.2	-0.2	99	99
LIMIT-4	4	0.0	-0.5	-0.7	-0.2	-0.2	-0.2	103	98
NODDV	14	0.0	-0.3	-0.3	-0.1	-0.1	-0.1	97	99
ALLOWRED	14	0.0	-0.2	-0.3	-0.1	-0.1	-0.1	98	98
MEAN	14	0.0	-0.3	+0.1	0.0	0.0	0.0	100	98
NOAMVP	14	0.0	-0.6	-0.7	-0.3	-0.2	-0.2	101	98
APPLYRES	14	0.0	-0.6	-0.8	-0.3	-0.2	-0.2	108	98

Table 10: Average BD-Rate coding results for the different variants

tained in the original method but alleviate the median computation in hardware in the worst-case scenario. The NODDV, ALLOWRED and MEAN variants however keep the same *MaxCand* as in the original method but significantly reduce the gains (losses are even reported for the MEAN variant on the second dependent view). Finally, the NOAMVP variant slightly reduces the gains while not affecting runtime, while the APPLYRES variant, on the contrary, achieves the same coding performance as the original method with the same *MaxCand* but with an increase in encoder runtime (108%).

Visual results – The significant gains on the dependent views for the Kendo and Dancer sequences in the proposed method are visible in Figure 16. Parts of the left view (view 1) and the right view (view 2) at a QP of 40 and 35 for the Kendo and Dancer sequences respectively, coded using the HTM-5.0.1 reference software and with the proposed method are shown in this figure. For the Kendo sequence, we can see that our method avoids having the sword broken in two as in the reference. For the Dancer sequence, the back of the dancer's head is more sharply represented using our method.



(a) Kendo V1 QP40 reference (b) Kendo V1 QP40 proposed (c) Dancer V2 QP35 reference (d) Dancer V2 QP35 proposed

Figure 16: Parts of dependent views coded with the reference software and with the proposed method

4.3.3 Results interpretation

Origin of the gains – The proposed method improves the quality of the DV used in IVMP. Consequently, the multi-view and the inter-view candidates in the Merge list, which depend on that DV, are also improved and more often selected. Table 11 shows the increase in the number of PUs coded in Merge mode using the multi-view or the inter-view candidates, in the proposed method, for each tested sequence, averaged across four QPs. A significant increase is noted for the inter-view candidate (31% on average) since it directly corresponds to the improved DV. For the multi-view candidate, the improved DV is only used to find a PU in the base view from which to extract a MV. Consequently, the improved DV may point to a PU that has the same MV as the one of the PU pointed to by the original DV. In this case, our DV improvement has no effect, and this explains why on average, the selection of the multi-view candidate has only slightly increased (2%). In any case, these increases are directly correlated with the coding gains achieved using our method.

Sequence	**Multi-view increase**	**Inter-view increase**
Kendo	0.6	23.3
Newspaper	1.4	21.5
Balloons	3.6	18.4
Dancer	3.5	54.8
GT Fly	0.5	62.9
Poznan Hall2	2.2	16.7
Poznan Street	1.4	20.1
Average	**1.9**	**31.1**

Table 11: Increase in the percentage of PUs coded in Merge mode using the multi-view or the inter-view candidates

Runtime results analysis – Furthermore, Table 12 shows the average, minimum and maximum number of vectors on which the median is computed for each tested sequence in the encoder and the decoder, in the proposed method. We can see that the worst case scenario in which the median is computed on 14 values never occurs for

any sequence (maximum is 12). On average, the median is computed on 1.9 vectors at the encoder and 2.2 vectors at the decoder, the difference being due to the fact that the encoder tests all possible CU sizes and partitions and hence performs the median computation much more often than the decoder. In any case, most of the time, the median computation is simple and is performed quickly. This explains why the runtime increase at both coder sides was imperceptible. Indeed, this increase definetely exists since our method necessarily adds some operations to the encoder and decoder without removing others, but it is not visible in Table 9 because it is really small.

Sequence	**Encoder**			**Decoder**		
	Avg	**Min**	**Max**	**Avg**	**Min**	**Max**
Kendo	1.9	1	11	2.2	1	9
Newspaper	1.9	1	11	2.1	1	10
Balloons	1.9	1	10	2.2	1	10
Dancer	1.9	1	10	2.2	1	10
GT Fly	2.3	1	12	2.4	1	12
Poznan Hall2	1.7	1	11	1.9	1	10
Poznan Street	2.0	1	11	2.2	1	11
Overall	**1.9**	**1**	**12**	**2.2**	**1**	**12**

Table 12: Average, minimum and maximum number of vectors for median computation at the encoder and decoder side

Variants results interpretation – The 1REF, 1REF+RMPOS and the LIMIT-4 variants all succeed in reducing *MaxCand* with a small penalty on coding gains (0.1% on coded texture videos). The performance of these three variants is roughly equivalent, but LIMIT-4 reduces *MaxCand* the most (to 4 instead of 9 or 7), making it clearly the best variant in this category. Note that Table 10 shows that LIMIT-4 increases the encoder runtime (103%) but as previously said, there is a ±3% error margin on this runtime so any increase below 103% or any decrease above 97% is not considered valid.

The gains are more significantly reduced in the ALLOWRED variant, in which the redundancy check on the vectors before median computation is not performed. This can be explained by the fact that the redundancy check allows to diversify the input vectors for the median computation, hence avoiding having the same DV chosen over a contiguous region with different disparity values. In addition, the redundancy check reduces the average and maximum number of vectors (considered on all sequences) on which the median is computed. Indeed, our experiments show that these values would have increased to 4.0 and 14 at the encoder, and 4.8 and 14 at the decoder, respectively, if the check was not performed.

Storing only the DVs in the list while discarding DDVs also reduces the gains of our method. Indeed, not considering DDVs in the list penalizes our method in case there are no DVs to insert. Indeed, in that case, the final DV used for IVMP is set to the zero vector, while in the reference method, a DDV may be chosen, which is almost always

more accurate than the zero vector. This result also validates our intuition discussed in Section 4.2.2 about the fact that the heterogeneity in lists in a video coder is more efficient than homogeneity.

If we do not apply our method for AMVP, the multi-view and the inter-view candidates in the AMVP list are not improved. Consequently, a slight reduction of the gains on the dependent views (0.1% loss) is noted with practically no influence on encoder runtime. This validates our assumption that the contribution of improving the multi-view and inter-view AMVP candidates in the proposed method is small.

If our method is applied for IVRP as well as IVMP, as in the APPLYRES variant, the coding gains would remain the same on average. This is because improving the multi-view and the inter-view Merge candidates has a much higher impact than improving IVRP. However, the slight increase in encoder runtime in our method becomes multiplied by around 2.5 since IVRP is applied for all PUs, including PUs coded in Intra, as opposed to IVMP. As a consequence, it becomes visible as seen in Table 10. Hence, for a better coding efficiency / complexity tradeoff, our method should not be applied for IVRP.

Finally, we have tested replacing the median computation with a simpler average computation (the MEAN variant). However, the coding gains obtained are small. Some losses are even reported for the second dependent view. This can be explained by the fact that the median allows to select a DV out of accurate, previously estimated DVs, whereas the average creates a new DV which might not truly describe the disparity at the level of the current PU.

4.4 Conclusion and future work

In this section, we have presented a method that tackles the sub-optimality problem in the current DV derivation process in 3D-HEVC (NBDV) resulting from selecting the first DV or DDV found in the search. In our method, all found DVs and DDVs in spatial and temporal neighboring PUs are stored together in a single list, and the search process is never stopped. Redundant vectors in the list are removed, and the median of the remaining vectors is computed and set as the final DV used for IVMP. Average bitrate reductions of 0.6% and 0.8% on the two dependent views, along with 0.2% on synthesized views were achieved with no increase in encoder and decoder runtimes. Several variants were tested as well, in order to either reduce the worst-case maximum number of vectors on which the median is computed, or to provide informative results.

The selection of the final DV can also be based on an R-D check applied on the candidates stored in the list. The DV selected would be the one yielding the lowest R-D cost. This requires sending the index of the DV in the list to the decoder, but the method might still bring significant gains.

5 Using elastic deformation on curves to encode depth maps

The main idea of this contribution is to exploit the tool of elastic deformation on curves [52], in order to have an effective predictor for lossless coding of countours. Since contours are the most important information in a depth map, this tool could be very useful in the contetx of MVD compression.

The input of the system consists in N depth images. They can be images from the same view at N temporal moments; or images at the same time from N different views; or they can be images from N_1 cameras at N_2 time instants (with $N = N_1 N_2$). Let $D^{(v,k)}$ be the depth image from view v at time k.

The depth images are segmented. For each image $D^{(v,k)}$ we obtain $M(v,k)$ lines, representing the contours of the objects. The contours can be open or closed. A depth image $D^{(v,k)}$ is then represented by:

- A set $\mathcal{C}(v,k)$ of $M(v,k)$ curves. Let $c_i^{(v,k)}(t)$ be the parametric representation of the i-th curve. We will drop the superscripts when there is no ambiguity: $c_i(t)$. When not necessay, we will drop the dependency from parameter t: c_i.
- The "interior" of each curve (ill-defined if the curve is open), *i.e.* the depth values of the object delimited by the contour.

Some examples of this kind of data is in Fig. 5, where the depth from the popular "ballet" sequence are shown. In Fig. 5 we show the associated contours, extracted with a very simple edge detector like the Canny-Deriche filter.

In the following we consider a first simplification: as in the examples, the depth images are made up only of a main object and the background. Therefore we have only a curve in the set $\mathcal{C}(v,k)$, *i.e.* $\forall v,k\ M(v,k) = 1$. Therefore we drop the subscript i. In this contibution we show some idea and results about the coding of the curves. This contribution could be integrated to the depth map compression method based on diffusion equations and shown in the previous deliverable (D4.2, Section 3).

5.1 Mono-dimensional compression

In this section we consider a first case of study. We have n curves, representing an object shape in a set of n depth maps. This set of curves is *monodimensional* in the sense that we either fix the camera and consider n temporal instants, or fix the time and consider n views. The proposed method could be extended to a bi-dimensional setup.

As previously noted, we consider only the simple case where there is only one object per image. The curve representing the object shape in the depth image related to time k and camera v is then $c^{(v,k)}(t)$.

Using the notation introduced in the previous section, here we consider one of the two sets of curves:

$$\mathcal{T}_k = \left\{c^{(v,k)}\right\}_{v\in\{0,\ldots,N\}} \qquad \mathcal{V}_v = \left\{c^{(v,k)}\right\}_{t\in\{k_0,\ldots,k_0+K\}}$$

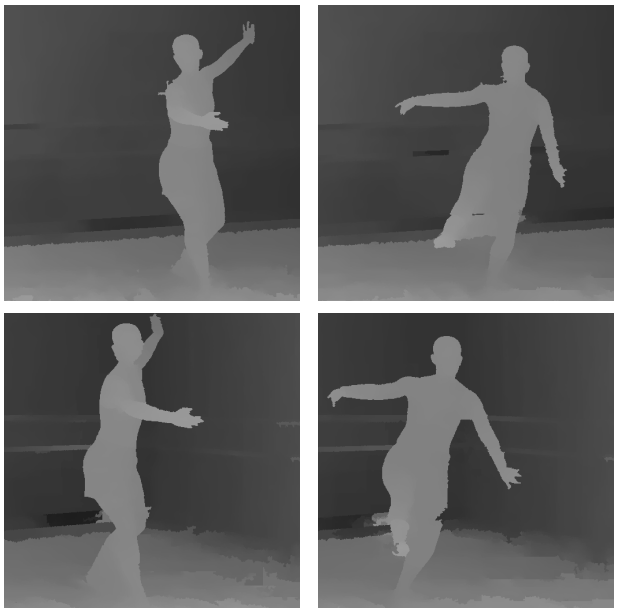


Figure 17: Depth images. Top line: view 1, $T = 1$ and $T = 5$; Bottom line: view 7, $T = 1$ and $T = 5$;

In order to have a uniform notation, we will call $\mathcal{U}$ the set of curves and $c^{(i)}$ the generic curve. As an example, in Fig 19, we used $\mathcal{U} = \mathcal{T}_k$ for the sequence ballet, with $k = 2$. In this figure, the red curves are $c^{(0)}$ and $c^{(N)}$, that is, the shapes of the object at the two farthest cameras (0 and 7). The blue curves are the original shapes at intermediate views, $\{c^{(i)}\}_{i\in\{1,\dots,N-1\}}$. The black line represent the geodesic between $c^{(0)}$ and $c^{(N)}$, and it is sampled in $N-1$ points. The black curves, $\widehat{c}^{(i)}$, are therefore the interpolated shapes.

In this scenario, we suppose that the decoder disposes of $c^{(0)}$ and $c^{(N)}$. This is a reasonable hypothesis, since it corresponds to a high quality encoding of depth maps $D^{(k,0)}$ and $D^{(k,N)}$. As a consequence, the decoder is able to compute the interpolated curves $\widehat{c}^{(i)}$.

Now, let us consider the decoding of the i-th shape. We propose this algorithm.

1. The encoder produces the geodesic α between $c^{(0)}$ and $c^{(N)}$
2. The encoder finds the best approximation of $c^{(i)}$ on α. Let us call it $\widetilde{c}^{(i)}$.
3. The encoder send the (quantized) coordinate of $\widetilde{c}^{(i)}$ on the geodesic. It amounts to a (quantized) real number in $[0, 1]$.

Let us define some details about these points.

As far as the geodesic generation (point 1) is concerned, according to our hypotheses, the encoder and the decoder can generate exactly the same trajectory, since they both dispose of the same curves $c^{(0)}$ and $c^{(N)}$.

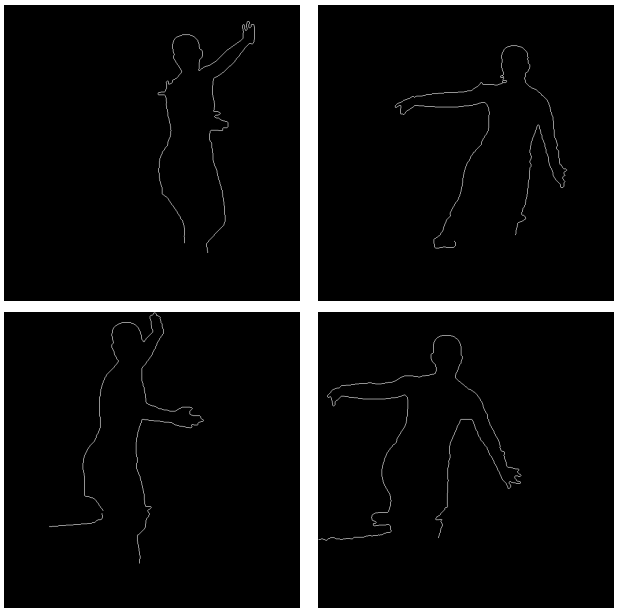


Figure 18: Contours extracted from the depth images. Top line: view 1, $T = 1$ and $T = 5$; Bottom line: view 7, $T = 1$ and $T = 5$;

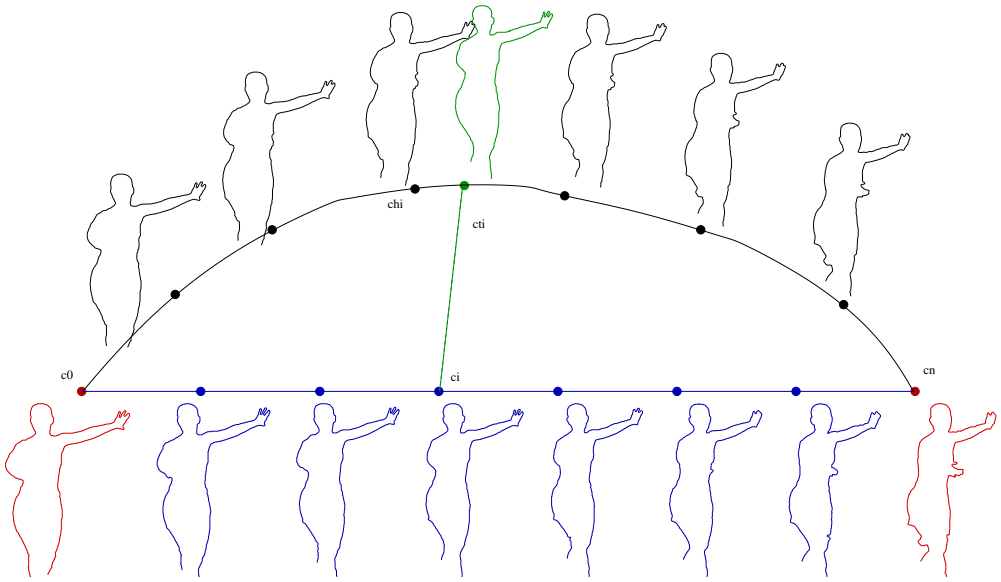


Figure 19: Actual shapes (blue) and interpolated shapes (black). The best shape on the geodesic is in green

We look for the best curve on the geodesic for the representation of $c^{(i)}$ (point 2). A first solution is to simply use $\widetilde{c}^{(i)}$. This would cost 0 bits, since the decoder can generate $\widetilde{c}^{(i)}$ for any i. However, it is possible that a better representation of $c^{(i)}$ exists on the geodesic.

However, both the encoder and the decoder share the same prediction of the current contour. This information can be used in order to enhance the context of an arithmetic encoder representing the shape. More precisely, the contour can be represented as a sequence of symbols: this is the "chain coding" format. In the chain coding representation, a contour is a sequence of three symbols, accounting for "go straigth", "turn left", "turn right". A context-based arithmetic encoder can easily take advantage of the high-order correlation among these symbols, as shown in [15]. We improve upon this method by using the elastic deformated curve as further conditioning state for the arithmetic encoder. Our tests show that, using 8 symbols on the elastic curve around the current position allow to correct the symbols probabilties and therefore to achieve a rate reduction of about 6 %.

6 Layered Depth Video coding using MV-HEVC

6.1 Introduction

Layered Depth Image/Video (LDI/LDV) representations are attractive compact representations for multi-view plus depth video sequences. Any virtual viewpoint can be rendered from LDV by using view synthesis technique. However, rendering from classical LDV leads to annoying visual artefacts, such as cracks and disocclusions.

This work aims at comparing the use of improved I-LDV (incremental way of building LDVs) and MVD while using MV-HEVC for encoding them. The remainder of this section is organized as follows. The MVD and LDV representations are firstly described and analyzed. The construction method of I-LDV is first presented. The problems of sampling and ghosting artefacts are tackled via the introduction of dedicated inpainting, depth discontinuity detection, followed by a local foreground / background classification.

6.2 Multiview plus depth and Layered depth video representation

The multiview video plus depth (MVD) format has been introduced in order to allow for flexible view synthesis and rendering on the terminal. In the MVD format, the multiple video sequences captured for the same scene are accompanied by the corresponding per-pixel depth map sequences. When used together with depth-image-based rendering (DIBR) algorithms, MVD sequences allow the generation of virtual views of the scene from any viewpoint [10, 69]. This property can be used in a large variety of applications [51], including Three-Dimensional TV (3DTV), Free Viewpoint Video (FTV), security monitoring, tracking and 3D reconstruction. However, the MVD contents generate very large amounts of data which motivates the design of dedicated efficient compression algorithms [38]. The compression algorithms very much depend on the input data representation, which can directly be the MVD format or some intermediate more compact representations. The data representation, the compression algorithm and the view synthesis method used on the rendering side, are very much dependent from each other and have a strong impact on the trade-off between compression efficiency and view rendering (including virtual view rendering in a context of FTV) quality.

The LDI representation [49, 64] is one particular Depth Image Based Rendering approach. In this representation, pixels are no more composed by a single color and a single depth value, but can contain several colors and associated depth values. The LDI extends the 2D+Z representation, but instead of representing the scene with an array of depth pixels (pixel color with associated depth values), each position in the array may store several depth pixels, organized in layers. This representation is illustrated in Figure 20. This representation reduces efficiently the multi-view video size, and offers a fast photo-realistic rendering, even with complex scene geometry. Various approaches to LDI compression have been proposed [18, 64, 65], based on classical LDI's layers constructions [13, 64]. However, the layers generated are still correlated, and some pixels are therefore redundant between layers.

(a) 1st layer (b) 2nd layer (c) 3rd layer

Figure 20: First layers of an LDI frame. 8 inputs views are used for the generation. ($\Delta_d = 0.1$)

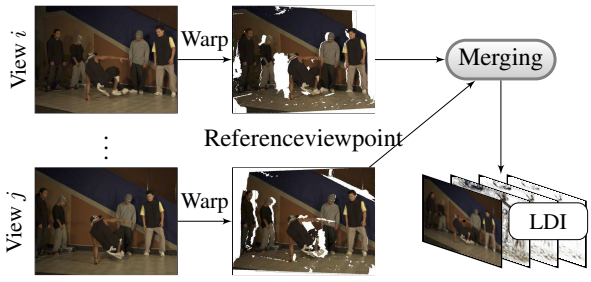


Figure 21: Step of I-LDI construction for view i, with residual information extraction.

6.3 Incremental Layered Depth Images

The section reminds briefly the algorithm for creating I-LDI, which has already been detailed in [1]. The reference viewpoint is one of the input viewpoints. Then, this I-LDI is warped iteratively on every other viewpoint (in a fixed order). The synthesized view is then compared with the original acquired view, and the discovered information is isolated, using a logical exclusion difference between the real view and the warped I-LDI to compute the residual information. This discovered information is warped back into the reference viewpoint and inserted in the I-LDI layers. By this method, only required residual information from side views is inserted, and no pixels from already defined areas are added to the L-LDI. On the other hand, all the information present in the MVD data is not inserted in the I-LDI. The first three layers of such an I-LDI are presented in Figure 22. Compared to LDI layers, I-LDI layers contain fewer pixels, and these pixels are grouped in connected clusters. Indeed, with this method, only required extra (or residual) information from side views is inserted, and no pixels from already defined areas are added to the I-LDI. On the other hand, all the information present in the MVD data is not inserted in the I-LDI, reducing the correlation between layers.

Finally, I-LDVs are direct extensions of I-LDIs as no further temporal processing is performed to build video layers. The following subsection briefly describes the MV-HEVC coder and the choices that have been made by the JCT-3V group.

(a) 1st layer (b) 2nd layer (c) 3rd layer

Figure 22: First layers of an I-LDI frame. All 8 inputs views are used for the generation, in a B-hierarchical order.

6.3.1 MV-HEVC coder

The **M**ulti**V**iew **H**igh **E**fficiency **V**ideo **C**oding scheme MV-HEVC is still under development by the Joint Coding Team JCT-3V (**H**EVC **T**est **M**odel version 5.1 at the time of writing [20]). At the Call for Proposal for a multiview version of HEVC, two main solutions were proposed: MVD and LDV. The first option has been chosen in which MVD content is directly encoded, based on HEVC [5] coding tools. In the MV-HEVC scheme, the temporal dependencies are extended to inter-view components by means of extra prediction tools. The encoder takes as input:

- the videos captured at different viewpoint locations,
- their corresponding depth maps (optional)
- camera parameters for optional view synthesis at decoder side.

Figure 23 illustrates the encoder side with the different input and connections. One can notice that the reference view is independently encoded so that a device equipped with a HEVC decoder can display a monoscopic video.

The next section describes our test conditions and how MV-HEVC is used in order to encode LDV content.

6.3.2 Experiments

Experiments have been carried out in order to see the potential of LDV in terms of coding performance. In these tests, the ongoing coding tool MV-HEVC is used for encoding both MVD and LDV coding.

Using MV-HEVC for LDV coding

No changes have been done in the current MV-HEVC software to compress our LDV representation. The first layer is encoded as the reference view. The second layer is coded as a dependant view.

The problem of ordering views has come because we are trying to encode two views with the same parameters. In order to quickly solve this problem, a slight difference

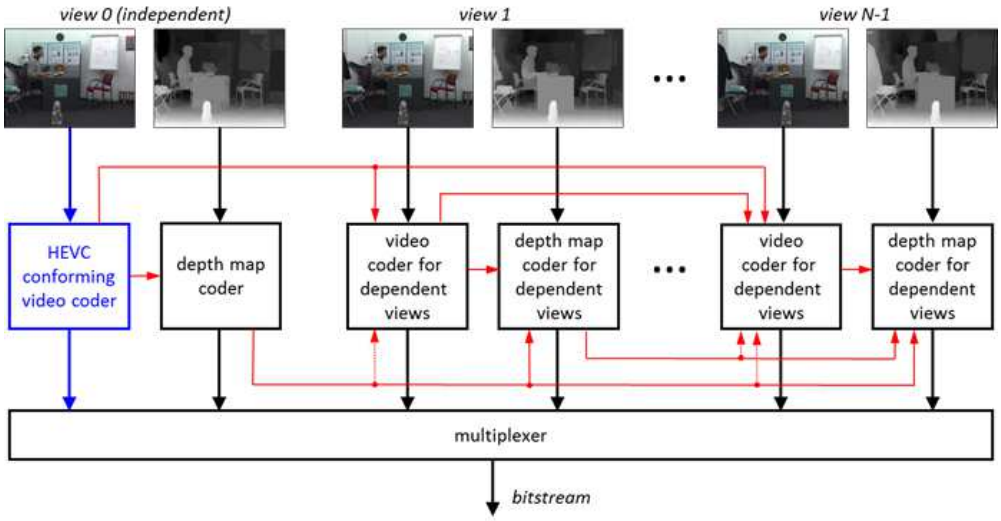


Figure 23: MV-HEVC encoder scheme. View 0 is the reference view which can be independently decoded using a HEVC decoder.

in thr translation parameter has been added. The MV-HEVC parser can then order the two views even if there is no need to.

MPEG test conditions

The MPEG Common Test Conditions (CTC) aim at showing the performance of the MVD scheme. Two major outputs are tested:

- the output encoded viewpoints,
- some intermediate virtual views synthesized from videos and depth maps provided in the bitstream.

In the first case, the performance can be classically assessed using objective metrics such as PSNR or using subjective tests. For synthesized views, the final virtual view build from decoded views is compared to the synthesis of the same viewpoint from the source input views. The set of test video viewpoints to be encoded have been chosen so that a stereoscopic pair can be build with a decoded view and an intermediate virtual view. Stereoscopic subjective tests can then be performed from synthesized views or both decoded and synthesized views.

Our test conditions

In the context of comparing two representations, the quality of synthesized views cannot be assessed by comparison with a synthesis from source views. Indeed source content would be different for MVD and LDV synthesis. Final synthesized content

Video	*Viewpoints*
Balloons	**1** 2 **3** 4 **5**
Newspaper	**2** 3 **4** 5 **6**
Dancer	**1** 3 **5** **9**

Table 13: Viewpoint numbers used for the three tested sequences. Balloons and *Newspaper* are of resolution 1024 × 768, Dancer is of resolution 1920 × 1088. Figures in bold correspond to encoded viewpoints.

is thus compared to some intermediate acquired views. For example, the *Balloons* sequence contains the viewpoints 0, 1, 2, 3, 4, 5. In the MPEG CTC, viewpoints 1, 3, 5 are encoded. This enables viewpoints 2 and 4 to be used for such comparisons. Table 13 presents the videos used in our tests and the corresponding view points.

The MVD content is encoded/decoded using the 3 view case in the MPEG common test conditions [20]. Only the synthesis of the captured intermediate viewpoints are performed. The LDV is built from the same 3 viewpoints which are encoded in the MVD case. The middle viewpoint is the reference on which are built the layers. The first layer contains the pixels visible from the reference view. The second layer contains extra information visible from other viewpoints, warped onto the reference view. One can notice that information is lost since the LDV has the same size as captured views. Indeed, while warping discovered pixels, those on the borders can be warped outside the reference frame. In order to facilitate interview (interlayer in our case) prediction of MV-HEVC, pixels from the first layer are copied on empty positions on the second layer for both texture and depth content.

6.3.3 Results

These tests only aimed at showing how promising LDV can be. Indeed the LDV and MVD are compared using the same coding scheme HEVC-3D which has been developed and improved for MVD content.

Figure 24 shows the PSNR levels for a synthesized intermediate view with both LDV and MVD decoded content. One can see that the MVD approach outperforms LDV at every bitrate. The LDV curve seems to be less sensitive to bitrate variation. Indeed, the dramatically difference in PSNR comes from the synthesis of the borders. Figure 25 shows the same sequence, same viewpoint but the PSNR is computed on cropped videos in order not to take synthesized borders into account. One can first notice that the objective metrics gives much better results for the MVD representation.

The objective results in terms of SSIM [60] follow the PSNR curves. Figures 26 and 27 show the SSIM curves for the Dancer sequence. In the case of figure 26 the MVD curve corresponds to the decoded view 1 whereas this viewpoint is synthesized from the decoded LDV. MVD logically overcomes because no synthesis is performed. However, figure 27 shows that LDV is not over MVD curve, even if the two representations give close SSIM levels.

In terms of subjective assessment, no MOS have been computed but expert viewing

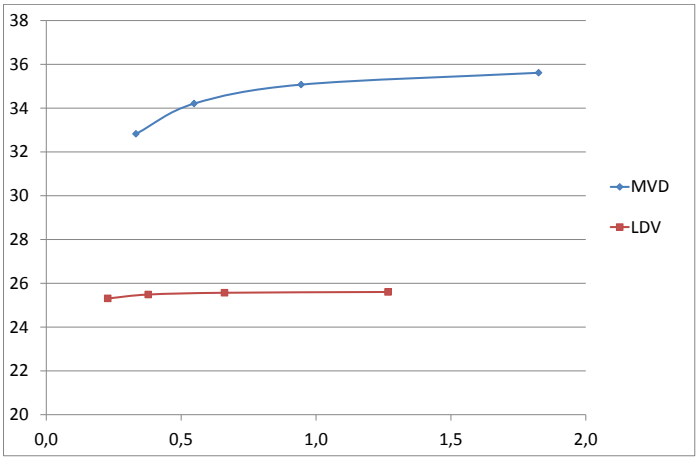


Figure 24: RD curves of the synthesis of viewpoint 2 of the *Balloons* sequence after compression of both MVD and LDV.

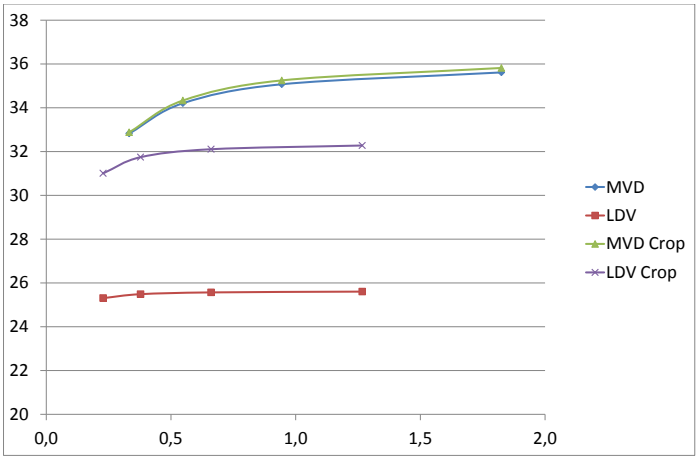


Figure 25: RD curves of the synthesis of viewpoint 2 of the *Balloons* sequence after compression of both MVD and LDV. In this case, frames are cropped to remove synthesized borders

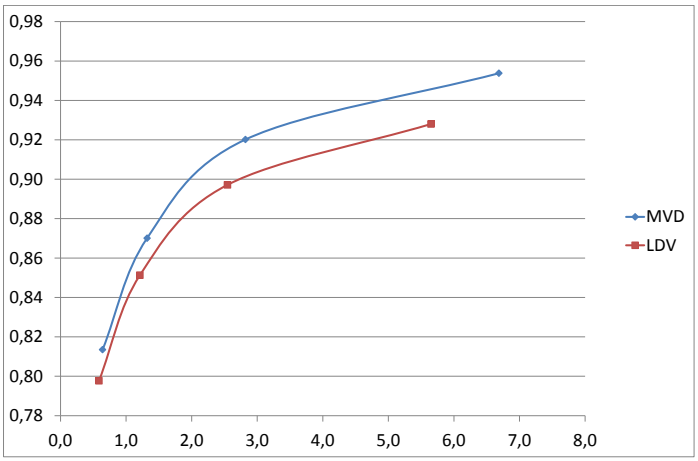


Figure 26: SSIM curves of the synthesis of viewpoint 2 of the *Balloons* sequence after compression of both MVD and LDV. Viewpoint 1 corresponds to a decoded view in the MVD case whereas it is synthesized from the decoded LDV.

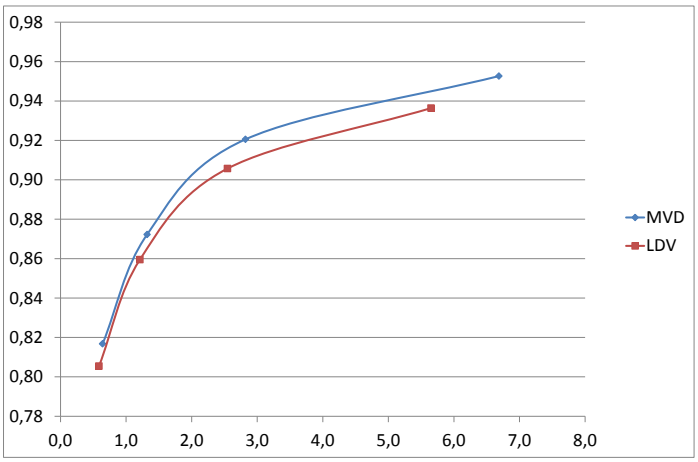
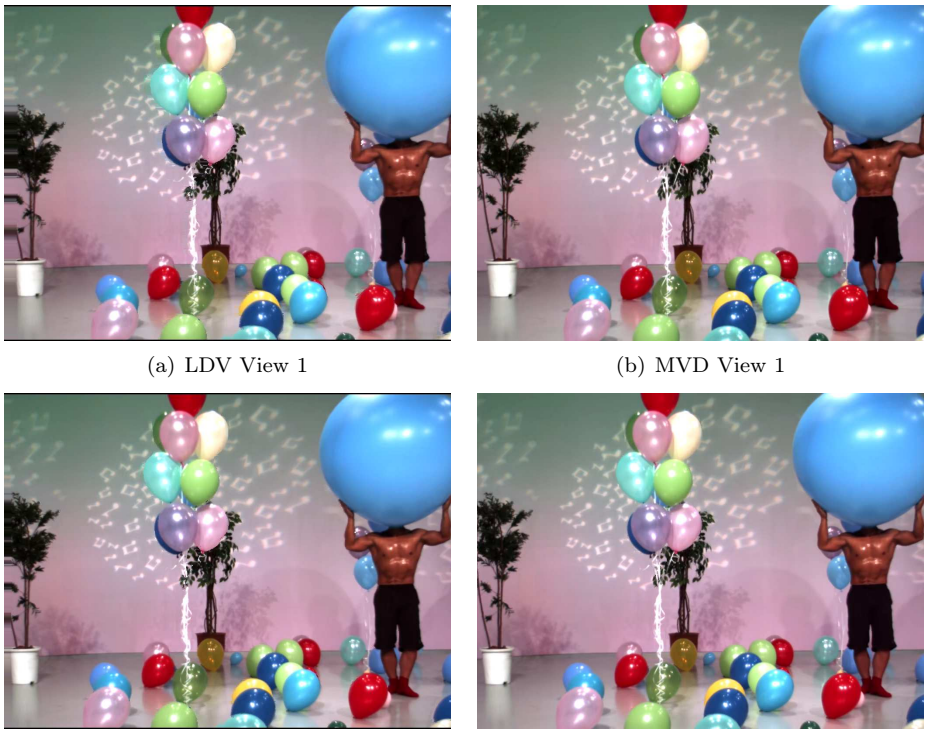


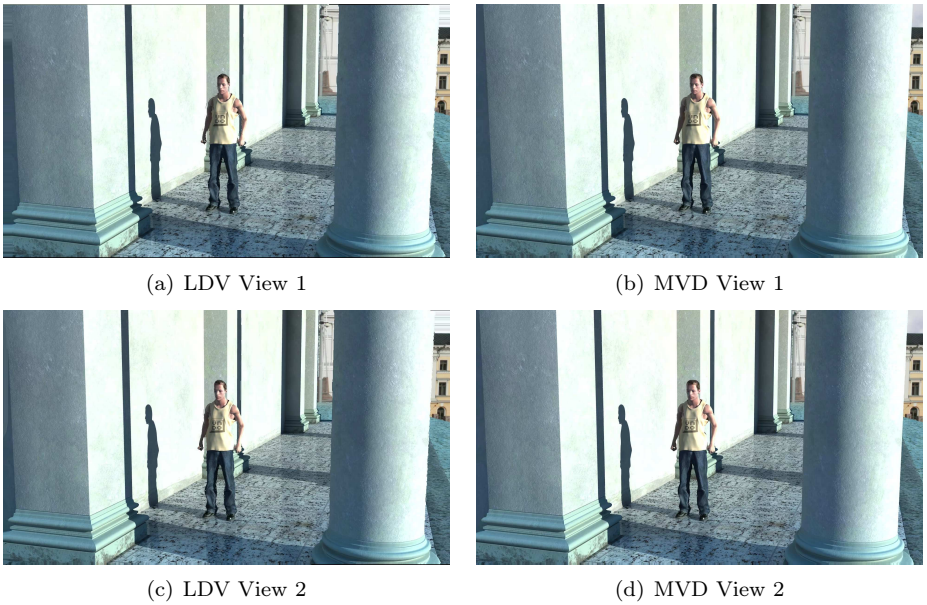
Figure 27: SSIM curves of the synthesis of viewpoint 3 of the *BDancer* sequence after compression of both MVD and LDV. Viewpoint 3 is an intermediate position which is synthesized for both representations.



(a) LDV View 1 (b) MVD View 1

(c) LDV View 2 (d) MVD View 2

Figure 28: *Balloons* Sequence encoded at $QP = 25$. Synthesis of viewpoint 2 with both representations.Viewpoint 1 corresponds to a decoded view in the MVD case whereas it is synthesized from the decoded LDV.



(a) LDV View 1 (b) MVD View 1

(c) LDV View 2 (d) MVD View 2

Figure 29: *Undo Dancer* Sequence encoded at $QP = 25$. Synthesis of viewpoint 3 with both representations.Viewpoint 1 corresponds to a decoded view in the MVD case whereas it is synthesized from the decoded LDV.

on final synthesized views suffice to conclude on the global lower quality of MVD content. Figure 28 shows the first images of the *Balloons* sequence. View 1 is just encoded and decoded in the MVD case so no synthesis artefacts appear, whereas the same view point has to be synthesized for LDV. One can notice the synthesized border which is missing in the LDV representation. This information, which is lost during the back warping of view 1 onto reference view (3), is not well reconstructed by texture synthesis because it relies on few confident data (this region is located on the frame border). Flickering also appear on the removed pixels for reducing ghosting artefact. Figure 29 provides another example with the *Undo Dancer* sequence.

Figure 30 shows viewpoint 3 reconstructed from both representations bitstreams at $QP = 40$. The synthesis artefacts are less sensitive because of the other classical coding artefacts. This is in accordance with the objective metrics curves.

The synthesis of borders has a huge impact on PSNR curves. Moreover, LDV synthesis lead to artefact on occluded areas from reference viewpoint. Even using a ghosting removal algorithm does not provide a perfect integration of pixels from other layers. In these MPEG conditions,

- low baseline
- interpolation only (synthesis of intermediate views)

(a) LDV (b) MVD

Figure 30: *Balloons* Sequence encoded at $QP = 40$. Synthesis of viewpoint 3 with both representations.

- rectified views
- coder designed and improved for MVD coding

LDV cannot fight. However, it remains a promising representation in other use case: extrapolation, high baseline, non parallel cameras.

6.4 Conclusions and future work

The MV-HEVC has been designed for MVD content encoding. Our tests show that MVD outperforms LDV in this context. However, is has to be reminded that LDV are used as input for a MVD dedicated coder. The test conditions are restricted to the interpolation case, so final synthesized viewpoint are located between source views. Moreover the views are rectified in the set of sequences which means that interview prediction without warping is powerful. Despite this context favouring the MVD approach, it appears that some crucial issues have to be solved in order to make LDV able to compete with MVD in such use cases.

- The left and right borders that are removed when warping side views onto the reference. This results in wide synthesized regions on one side of the synthesized view. Synthesis algorithm are not efficient enough to provide satisfying results on these side regions, in a temporal context.
- The way of handling ghosting artefacts relies on a spatial texture synthesis. Texture synthesis leads to poor results in terms of PSNR. Even for subjective observation, this synthesis results in temporal flickering on depth discontinuities.

Some future work can complete this study: different use cases can be considered in order to show the limitation of MV-HEVC common test conditions. LDV have to be improved whether by integrating a temporal texture synthesis or by handling borders.

7 Depth fading: a strategy for enhancing visual quality of low-bit-rate encoded 3D Videos

7.1 Motivations

Our goal is to achieve depth maps compression because up to now, there is no standardized compression method for MVD sequences. However, MPEG is currently standardizing a novel MVD encoding framework, namely 3DVC. Most of the proposed compression methods rely on the extension of state-of-the-art 2D codecs. The most popular is H264/AVC [54] whose 3D extension (standardized for Multi-View-Video representation, MVV), namely H.264/MVC for Multi-view Video Coding [37], has been the subject of many adaptations for MVD compression [43]. Previous studies already pointed out the impact of depth encoding on the synthesized frames. Compression-related artifacts that may be imperceptible in depth maps cause important distortion during the synthesis process [36]. Many methods have been proposed recently in order to address the aforementioned issues. Various encoding strategies are possible to achieve depth map compression. Several studies have proposed bit-rate control methods [16, 42] relying on the objective quality of the resulting synthesized views, or on a distortion model [31]. A popular and efficient strategy is the post-processing of depth maps after decoding [17]. Depth-adapted encoding methods [22, 41, 47] have also been proposed. Our work is in line with the depth-adapted encoding strategy since the method proposed in this chapter relies on the content-based representation of the depth map of LAR codec.

We believe that depth map compression can be achieved with LAR codec tools by means of several changes in order to adapt the strategy according to depth maps specificities. The main purpose of this novel framework is to preserve the consistency between color and depth data. Our strategy is motivated by previous studies [36] of artifacts occurring in synthesized views: most annoying distortions are located around strong depth discontinuities and these distortions can be due to misalignment of depth and color edges in decoded images. Thus the method is meant to preserve edges and to ensure consistent localization of color edges and depth edges. The LAR codec is based on a quad-tree representation of the images. In this quad-tree, the smaller the blocks, the higher the probability of the presence of a depth discontinuity. Analogously, big blocks correspond to smooth areas. The quad-tree representation contributes in the preservation of depth transitions when target bit-rate decreases. Another original contribution of the proposed method relies on the use of the decoded color data as an anchor for the enhancement of the associated decoded depth, together with information provided by the quad-tree structure. This is meant to ensure consistency in both types of data after decoding. We also propose to change the quantization strategy so that the artifacts occurring in the rendered view are less perceptible or less annoying.

7.2 Depth map encoding method

Based on previous results, since depth maps do not contain high frequency areas, the details are not essential and represent an avoidable additional cost of compression. Thus, only the flat image is considered and encoded in the method we propose, i. e. we use

LAR Flat pyramid only profile.

7.2.1 Quad-tree resolution

The quad-tree decomposition is dependent on the local gradient of the depth image. Given a threshold Y for the local gradient, the image is split into blocks: the higher the local activity, the more splits. This leads to small blocks around object edges and bigger ones in continuous areas. In the original LAR method, the minimal size of the blocks, N_{min} is equal 2×2. In previous experiments, we observed that using $N_{min} = 1$ instead of $N_{min} = 2$ provides better visual results on the synthesized view, but increases the bit rate. Since our priority is to enhance the visual quality, we first opt for $N_{min} = 1$.

Pasteau *et al.* [44] suggested applying a quantization step depending on the block sizes, in the case of conventional images. Our experiments revealed that in the case of depth map compression, this was not an adequate strategy because the smaller the blocks, the coarser was the quantization (this allowed bit rate savings because small block are costly). Yet, small blocks correspond to strong depth discontinuities and errors occurring in these areas may have disastrous effect at the synthesis step. Figure 31a) shows the impact of the quantization as suggested in Pasteau *et al.* [44] (first column) at 0.06 bpp and using $N_{min} = 1$. Depth transitions are highly degraded and will result in errors in the synthesized frame (third column, crumbling artifacts around the head and around the legs of the chair). The synthesized frames obtained in Figure 31c) are generated from original color data and decoded depth maps in order to visually assess only the impact of depth quantization (i.e. not the combined effect of both color and depth compression) using Pasteau *et al.* quantization.

7.3 Overview of our proposed strategy

The method that will be presented in the following, namely Z-LAR-RP, differs on the prediction step. The minimal quad-tree block size is kept as 1×1.
The associated texture view can be encoded by any state-of-the-art color codec. The Z-LAR-RP uses the decompressed texture information to improve the prediction step involved in depth maps decoding. The compression scheme still relies on the pyramidal profile of LAR, previously referred as

LAR Flat pyramid only. In the previous proposed approach, the selection of the lowest level to be transmitted and decoded from the pyramid construction was not allowed. Yet, this option is available in the pyramidal profile of LAR codec for 2D color images. The method that will be presented is meant to overcome this limitation and to allow the selection of depth resolution and mainly to increase the performances

a) Decoded depth when applying quantization suggested in Pasteau *et al.*

b) Decoded depth when applying our proposed strategy.

c) Frame synthesized from decoded depth when applying quantization suggested in Pasteau *et al.*

d) Frame synthesized from decoded depth when applying our proposed strategy.

Figure 31: Comparison of two decoded depth maps at 0.06bpp, using the LAR method or the proposed method of rate control.

of the depth map coding framework (in terms of visual quality of the synthesized views and in terms of complexity).
Any level of the pyramid can be chosen as the lowest to be transmitted and the actual depth map size is reached thanks to a region-based prediction method, that will be presented in the following. Regarding rate control strategy and quantization, this method follows the principle such as the actual depth structure of the scene is modified when the bit-rate decreases. Fig. 32 gives an overview of the method.
In the following section, we show that the basic region-based segmentation method can be jointly used with decoded color data in order to improve the prediction step by propagating the decoded depth values in the smallest blocks of the quad-tree. Then, validation experiments show the performances of the proposed Z-LAR-RP method.

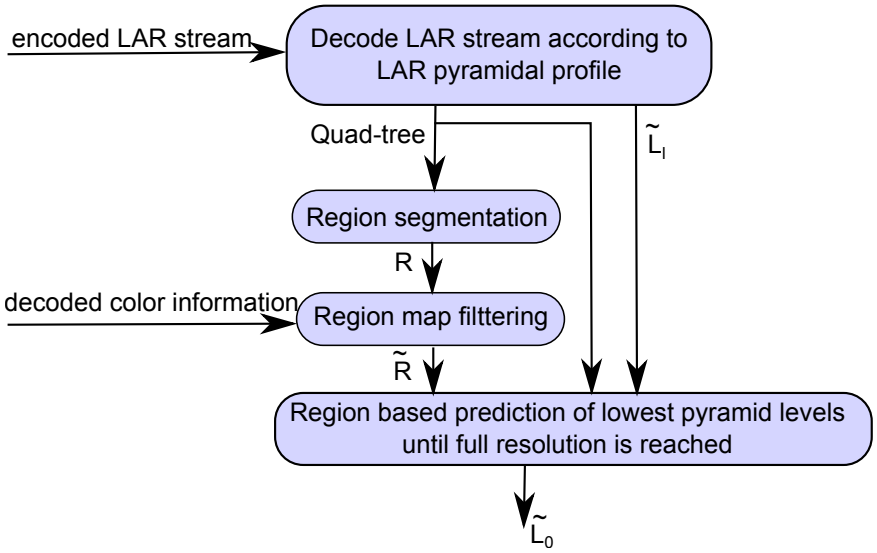


Figure 32: Overview of the Z-LAR-RP

7.4 Depth map encoding method

7.4.1 Region segmentation from decoded quad-tree

The region segmentation algorithm employed in this method, relies on previous work by C. Strauss presented in [53]. The specificity of this segmentation algorithm is that it only requires the knowledge of the image structure, that is contained in the quad-tree partioning of the image as input data. This quad-tree partioning is embedded in the very beginning of the LAR codec bitstream and can be extracted at the decoder side. Algorithm 1 gives the details of the segmentation algorithm as described in [53].

After creating the seeds from the larger blocks, adjacent regions are agglomerated by region growing. The process is reiterated $iter(CurrentSurf$ times. The number of growing iterations increases with the decrease of the size threshold $CurrentSurf$. Along the merging process, the number of regions should thus decrease. Fig. 33 depicts an example of the segmentation result. Fig. 33(a) is the quad-tree partition obtained from the first frame of *Breakdancers* depth map of camera 0, with $Y = 5$. Fig. 33(b) gives the first seeds, from the larger blocks. Fig. 33(c) gives the final region segmentation, with 370 regions.

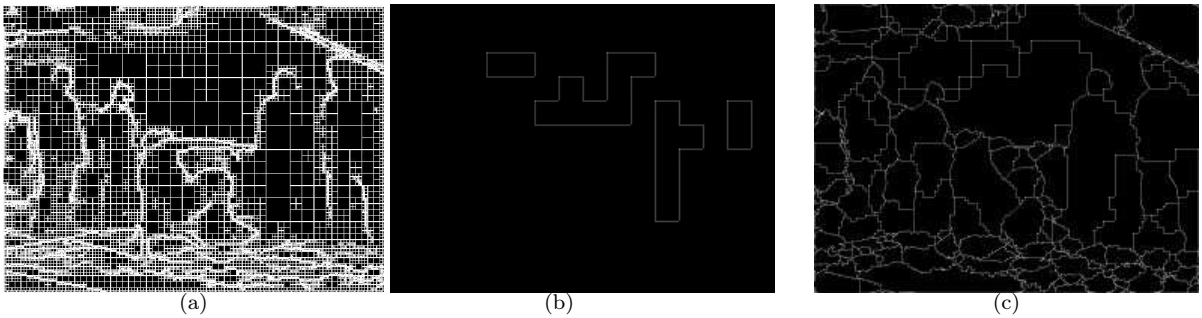
7.4.2 Color-consistent region edge refinement

In order to enhance segmentation results, we introduce the color information of the corresponding decoded color view. A discrete bilateral filter is applied on the region map obtained from the region segmentation process in order to refine the location of

Algorithm 1 Region segmentation algorithm from [53]

Require: **Quad-tree**$^{[N_{max}...N_{min}]}$ the dyadic quad-tree partition containing P square blocks b_i, $i \in \{1...P\}$ where each block b_i has a surface of $2^S \times 2^S$ pixels, $S \in \{1...N_{max}$;
Δ^k is the region map after k merging steps;
R_i^k in Δ^k is the k non overlapping region label;
$surf(R_i^k)$ is the surface in pixels of R_i^k;
A_i^k is the set of adjacent regions of R_i^k in Δ^k.
Initializations
$k = 0$
$\Delta^k =$ **Quad-tree**$^{[N_{max}...N_{min}]}$
$CurrentSurf = 2^{N_{max}} \times 2^{N_{max}}$
repeat
  Seeds creation
  while $\exists R_i^k | surf(R_i^k) = CurrentSurf$ **do**
    while $\exists R_j^k \in A_i^k$ and $surf(R_j^k) = CurrentSurf$ **do**
      Merge R_j^k and R_i^k into Δ^{k+1}
      $k = k + 1$
      Update A_i^k
    end while
  end while
  $CurrentSurf = \lfloor CurrentSurf/4 \rfloor$ {Region growing}
  while $\exists R_i^k | surf(R_i^k) = CurrentSurf$ **do**
    for iter=1 to $iter(CurrentSurf)$ **do**
      Let A' = $\{R_j^0 | R_j^0 \in A_i^k$ and $surf(R_j^0) = CurrentSurf$
      Let Z = card(A')
      Merge R_i^k and A' into Δ^{k+Z}
      $k = k + Z$
      Update A_i^k
    end for
  end while
until $CurrentSurf = 0$

decoded depth map edges to be consistent with color map edges. Algorithm 2 gives the details of the method. The region map is denoted R. Any pixel p at location (i, j) belongs to labeled region $R(p) = R(i, j)$ in region map. The filtered region map is noted as $\tilde{R}$.
For each pixel p, a support Γ_p is considered, that is the neighborhood of$p$, centered on p. The filter proceeds in way that Pixel p will be given the most likely region label according to the importance of its neighbors. This importance (or weight) of each neighbor is evaluated regarding its color similarity with p in the corresponding location in the decoded color image, and regarding its distance to p. Finally p is allocated the same region label as the neighbor having the highest importance (or



(a) Quad-tree partition, $Y = 5$ (b) First seeds (c) Region segmentation result with 370 regions

Figure 33: Region segmentation using [53]

weight)
The luminance component *Luma* of the decoded texture view is used to estimate the color similarity of the considered neighborhood. The algorithm 2 aims at assigning each pixel of the region map the more likely region label according to the criterion described earlier. These constraints are are expressed by the factors σ_c and σ_d respectively.
Fig. 34 depicts a snapshot of the result of this process over the region map, obtained with a neighborhood of 7×7 pixels, centered on the processed pixel, $\sigma_c = 30$ and $\sigma_d = 3$. The region frontiers in white are superimposed on the original corresponding color view. It can be observed that the segmentation is more consistent to color data.

7.4.3 Pyramid truncation

Any level l of the pyramid can be chosen as the lowest to be transmitted and the actual depth map size is reached thanks to the region-based prediction method described by Algorithm 3. Any pixel of coordinates (i, j) is denoted as p. $\tilde{L}_{l_{min}}$ is the lowest encoded level image of the pyramid, with $l_{min} \geq 1$. The block $b^N(i, j)$ is as described in Eq. **??**: $b^N(i, j)$ is a block of size N, located at (i, j) in the quad-tree partition.N is the block size as described in Eq. **??**.For each predicted pixel p in the magnified level, a support $\Gamma_{\lfloor \frac{p}{2} \rfloor}$ is considered, that is the pixel neighborhood in $\tilde{L}_{l_{min}}$, the lowest decoded level image, centered on the corresponding processed pixel $\lfloor \frac{p}{2} \rfloor$. K is a normalizing factor defined as:

$$K = \sum_{q \in \Gamma_{\lfloor \frac{p}{2} \rfloor}} \delta_p(q) \cdot e^{-\frac{\left\| \lfloor \frac{p}{2} \rfloor - q \right\|}{2\sigma_1}} \cdot e^{-\frac{\left\| \tilde{L}_l(\lfloor \frac{p}{2} \rfloor) - \tilde{L}_l(q) \right\|}{2\sigma_2}}, \tag{7}$$

where $\delta_p(q)$ is the existence function defined as:

$$\delta_p(q) = \begin{cases} 1 & \text{if } \tilde{R}(p) = \tilde{R}(q) \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

Algorithm 2 Region segmentation enhancement of the depth map based on the decoded color information

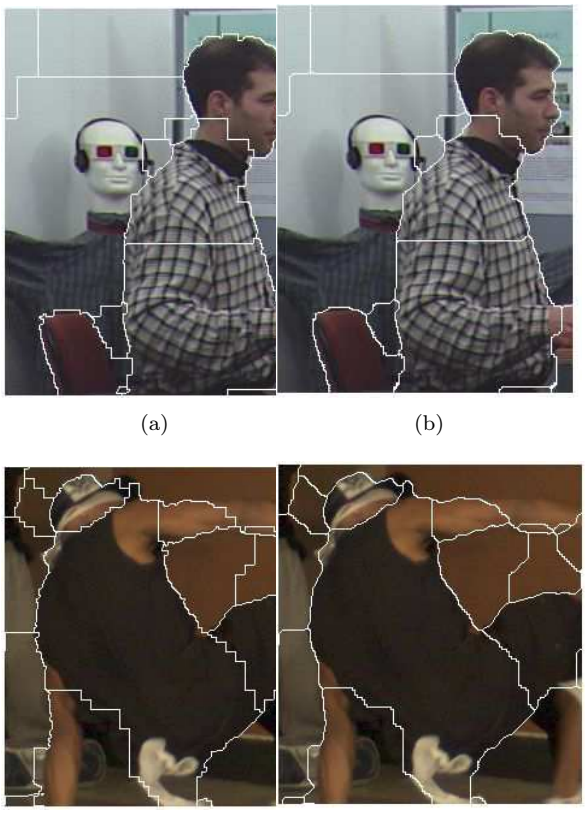
Require: R the region map of the depth image with $N_{regions}$ labels;
$W[N_{regions}]$ the array of region weights;
$Luma$ the associated decoded texture image
Initializations
$Temp(p) = Temp(i,j) = R(p) = R(i,j) \mid \{p = (i,j) \in N_x \times N_y\}$
$W[k] = 0 | \{k \in [1 \dots N_{regions}]\}$
for all $p \in R$ **do**
  for all $q \in \Gamma_p$ **do**
    $r = R(q)$
    $W[r] = W[r] + e^{-\frac{\|p-q\|}{2\sigma_d}} e^{-\frac{\|Luma(p)-Luma(q)\|}{2\sigma_c}}$
  end for
  Find $\tilde{r} \mid \tilde{r} = \underset{k \in [1 \dots N_{regions}]}{\operatorname{argmax}} W[k]$
  $Temp(p) = \tilde{r}$
  Reset all elements of W to 0
end for
$\tilde{R}(i,j) = Temp(i,j) \mid \{(i,j) \in N_x \times N_y\}$
return $\tilde{R}$

The reconstruction of depth lowest level images is based on a weighting sum of the corresponding neighbors in the direct upper level of the pyramid. The neighbors contribute into this weighting sum only if the belong to the same region in the full image resolution.

Algorithm 3 Region-based depth map prediction

Require: $\tilde{L_{l_{min}}}$ the depth lowest decoded level image of the depth map LAR pyramid with $l_{min} \geq 1$;
Quad-tree$^{[N_{max} \dots N_{min}]}$ the quad-tree partition;
$\tilde{R}$ the filtered region map.

repeat
  for all $p \in \tilde{L}_{l-1}$ **do**
    if $\tilde{L}_{l-1} \in b^N \mid N < 2^l$ **then**
      $\tilde{L}_{l-1}(p) = \frac{1}{K} \sum_{q \in \Gamma_{\lfloor \frac{p}{2} \rfloor}} \tilde{L}_l(q) \cdot \delta_p(q) \cdot e^{-\frac{\left\| \lfloor \frac{p}{2} \rfloor - q \right\|}{2\sigma_1}} \cdot e^{-\frac{\left\| \tilde{L}_l(\lfloor \frac{p}{2} \rfloor) - \tilde{L}_l(q) \right\|}{2\sigma_2}}$
    else
      $\tilde{L}_{l-1}(p) = \tilde{L}_l(\lfloor \frac{p}{2} \rfloor)$
    end if
  end for
until $l = 0$



(a) (b)

(c) (d)

Result of region segmentation

Region segmentation after enhancement process

Figure 34: Region segmentation after applying enhancement process

7.5 Experiment 1: objective quality assessment

7.5.1 Experimental protocol

The goal of these experiments is the validation of the Z-LAR-RP as an alternative to depth map coding. So only depth maps are encoded in order to highlight the impact of depth quantization strategies. Fig. 44 depicts the general scheme followed in these experiments. Depth coder under tests include the Z-LAR-RP, HEVC 6.1 and H.264 (JM 18) both in intra coding mode. The choice for these methods in this experiment is motivated by the fact that they are reference methods which are usually used as anchors in standardization process. Table 17 gives the details of the

quantization parameters used in these experiments. Six MVD sequences are used in these experiments: *Book Arrival*, *Newspaper*, *Kendo*, *Balloons* are real scenes; and *GT_Fly* and *Undo_Dancer* are synthetic scenes. Table 14 summarizes the sequences' features. The sequences were selected for their availability and amount of depth. The key frames were selected for their amount of depth. Table 18 gives the details of the encoded viewpoints and the target viewpoint for the synthesis. The synthesis process is performed throught the very last release of VSRS, that is the version used in MPEG 3DV group of standardization at the time of writing this thesis.

Sequence Name	**Resolution**	**No. of frames**	**Camera Arrangement**
Book Arrival	1024×768	100	16 cameras with 6.5cm spacing
Newspaper	1024×768	300	9 cameras with 5 cm spacing
Balloons	1024×768	300	7 cameras with 5 cm spacing, moving camera array
Kendo	1024×768	300	7 cameras with 5 cm spacing, moving camera array
GT_Fly	1920×1080	250	Computer generated imagery with ground truth depth data
Undo_Dancer	1920×1080	250	Computer generated imagery with ground truth depth data

Table 14: Six MVD sequences used in the experiments.

Sequence Name	**Encoded view points**	**View to synthesize**	**Frame no.**
Book Arrival	$10 - 6$	8	33
Newspaper	$2 - 6$	4	1
Balloons	$1 - 5$	3	1
Kendo	$1 - 5$	3	1
GT_Fly	$1 - 9$	5	157
Undo_Dancer	$1 - 9$	5	250

Table 15: Input and output views of the experiment.

Depth codec	**Quantization parameter**
H.264 (JM18)	Qp = [25, 27, 30, 33, 35, 37, 40, 42, 45, 47]
HEVC 6.1	Qp = [34, 36, 39, 41, 42, 43, 45, 46, 48, 50]
Z-LAR-RP	$Y = \{1$ to 241 $\}$, step by 10

Table 16: Input and output views of the experiment.

7.5.2 Results

Fig. 35, Fig. 36 and Fig. 37 depict the results of objective assessments through the widely used PSNR and MSSIM. However, the objective metrics are not sufficient to predict human perception of synthesized views quality, though MSSIM was one of the objective metrics giving the best results out of the tested set of metrics. Moreover, in the case of our proposed coding scheme, objective measurements based on the fidelity such as PSNR and MSSIM are inappropriate. Indeed, our coding method modifies the depth structure of the scene. Thus objects may be shifted. Since objective metrics are mostly FR, they measure the fidelity between two images and it is expected that our method obtain bad scores while having good visual quality performances. So we provide the PSNR of depth maps (average between the two views), the PSNR of the synthesized view, with the original acquired view as the reference and the MSSIM of the synthesized view, with the original acquired view as the reference, Fig. 35, Fig. 36 and Fig. 37, both as a rough guide. Snapshots of the corresponding views are provided in Fig. 38, 39, 40, 41, 42 and 43. Note that it can be observed a slight shift for Z-LAR-RP snapshots. The same viewpoint is always generated but at very low bit-rates, Z-LAR-RP tends to deliver a uniform depth map which results in a slight shift of the scene in the synthesized view. As expected the objective measures rate the Z-LAR-RP as worst than the two state-of-the-art codecs. This was expected because of the reasons mentioned above. However, visual analysis of all the synthesized views proves that the quality is often similar (Fig. 42) or even superior than that of state-of-the-art methods (Fig. 38, 39, 40, 41, 43). Moreover the proposed scheme allows very low bit-rates (around 0.003bpp). In these cases, the proposed scheme automatically transmit a flat depth map, which results in a good visual rendered view quality.

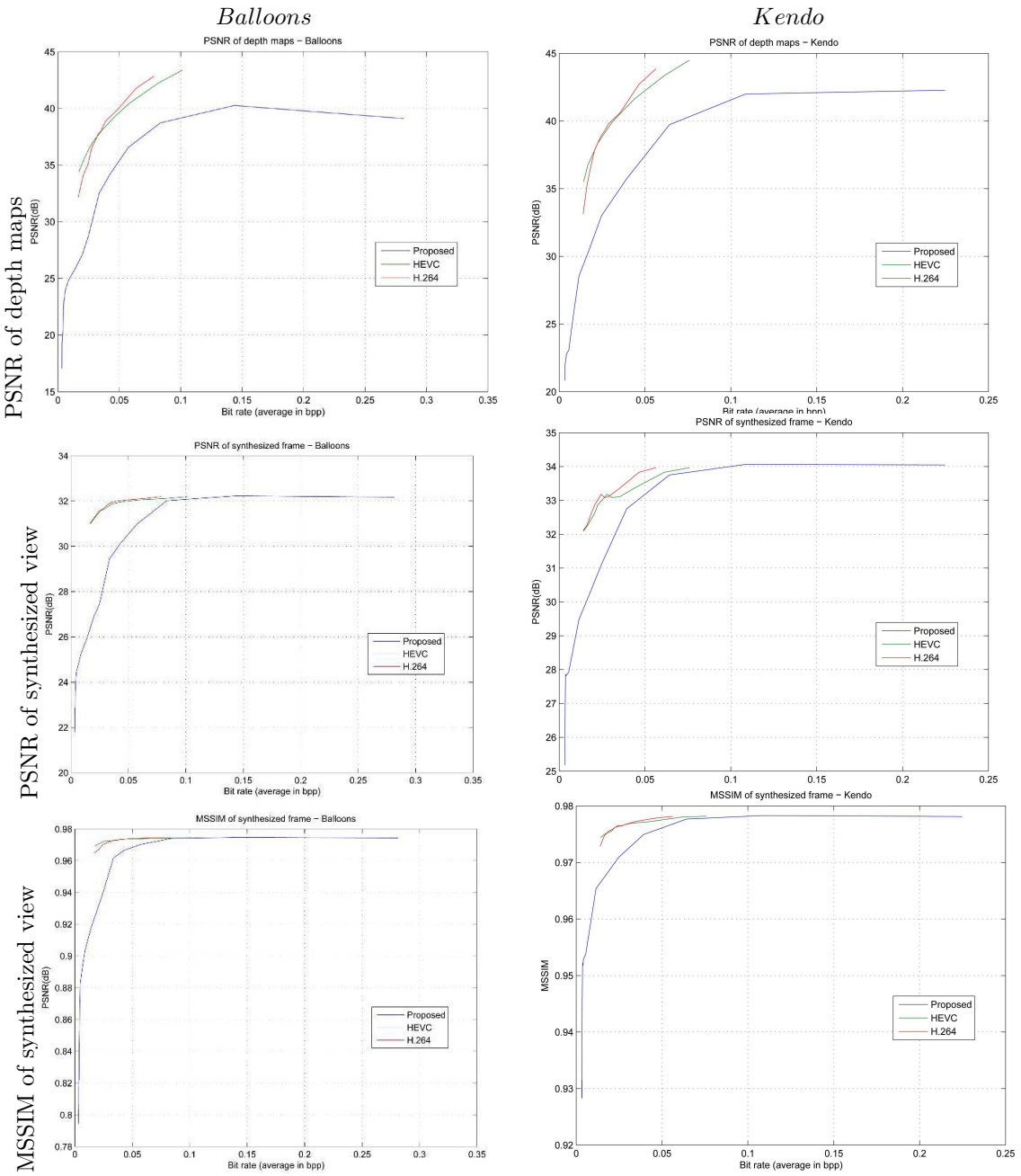


Figure 35: Rate/distortion curves of depth maps and synthesized views.

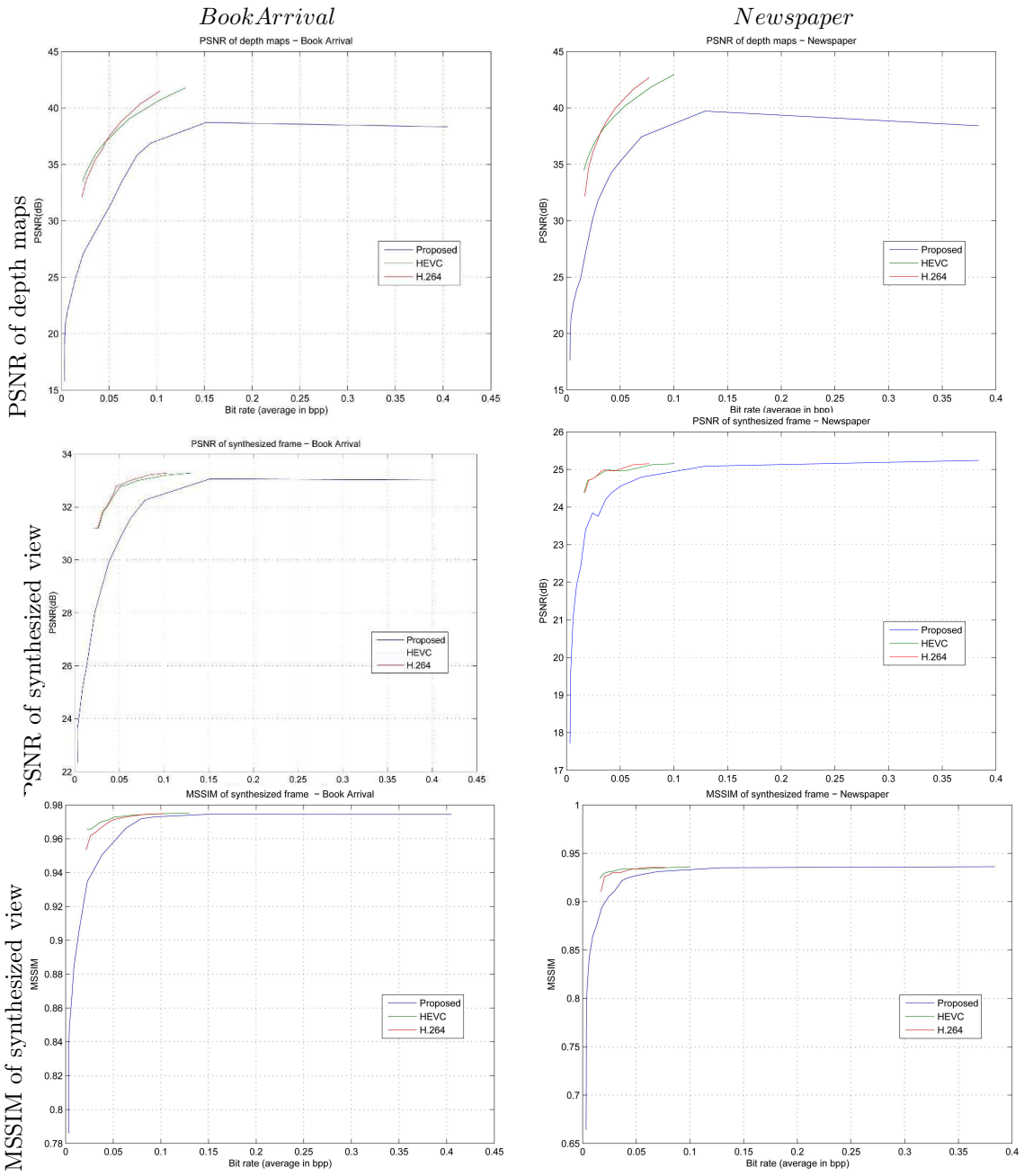


Figure 36: Rate/distortion curves of depth maps and synthesized views.

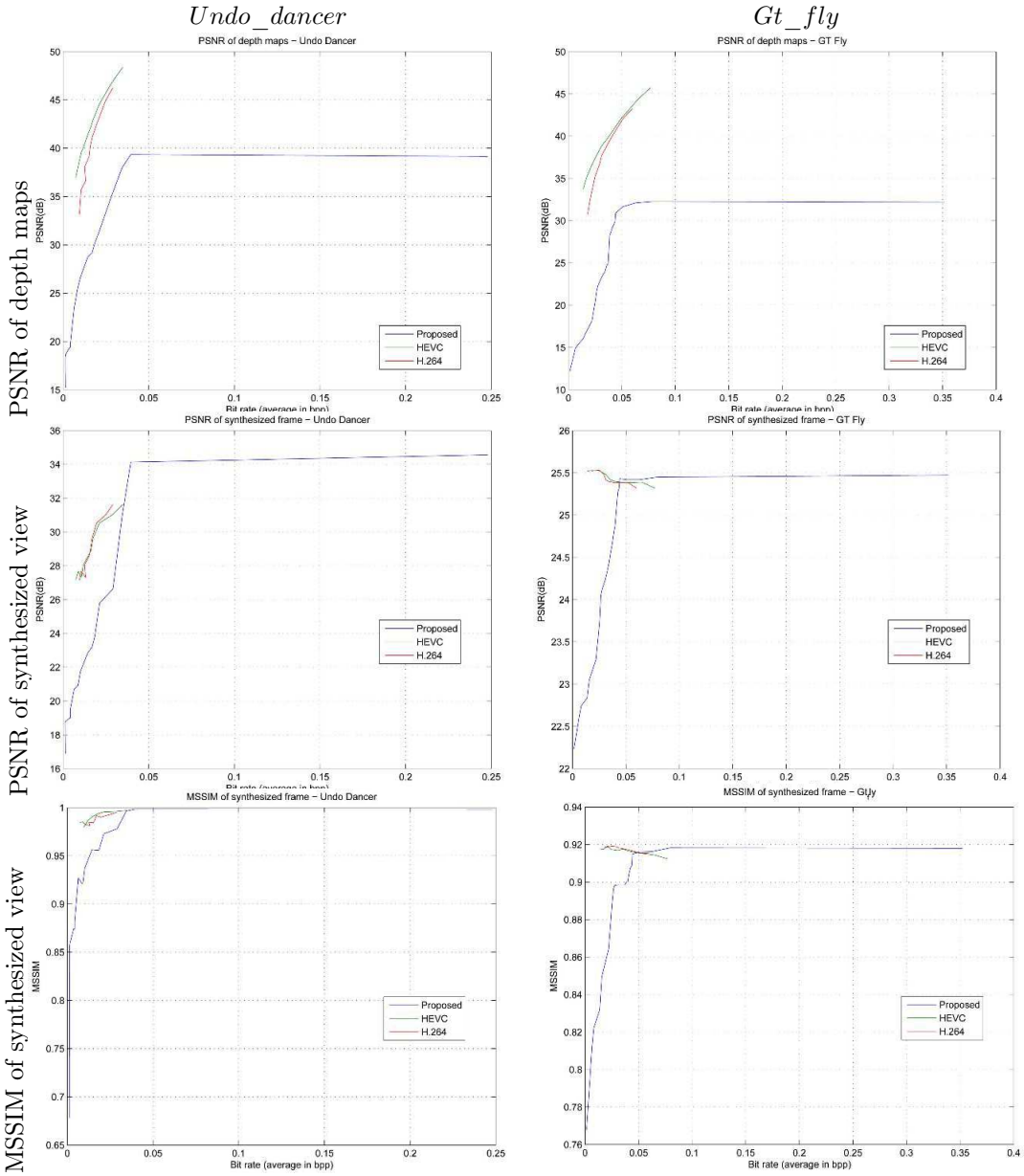
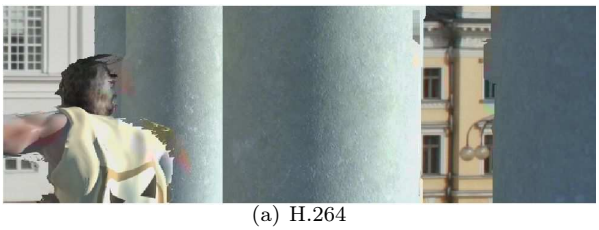
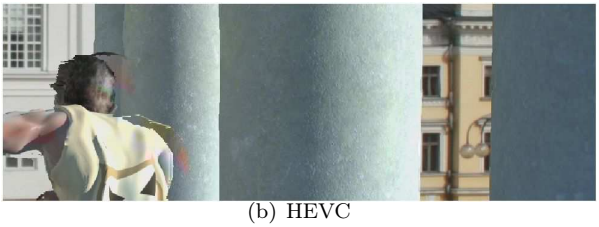


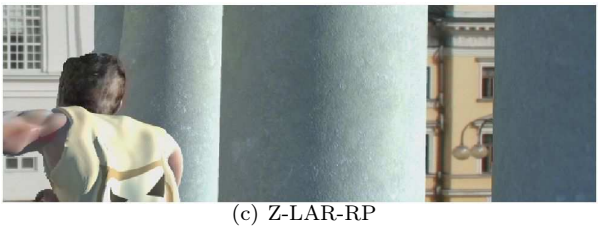
Figure 37: Rate/distortion curves of depth maps and synthesized views.



(a) H.264

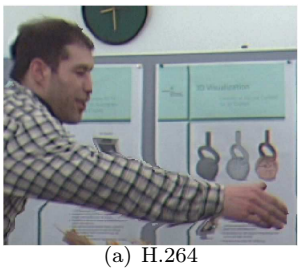


(b) HEVC

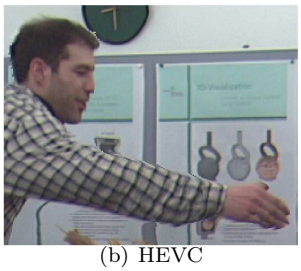


(c) Z-LAR-RP

Figure 38: Snapshot of synthesized frame - Undo_Dancer, 0.01bpp.



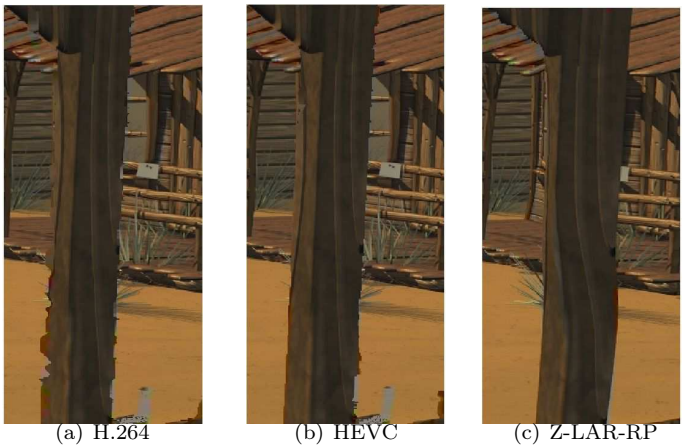
(a) H.264



(b) HEVC

(c) Z-LAR-RP

Figure 40: Snapshot of synthesized frame - Book Arrival, 0.02bpp.

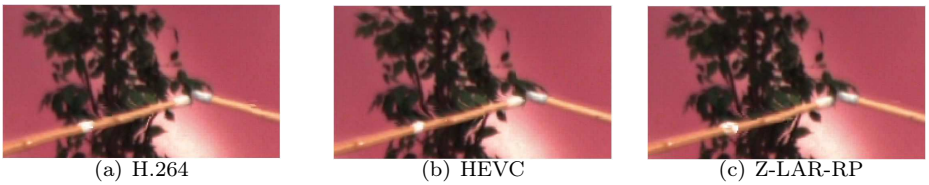


(a) H.264 (b) HEVC (c) Z-LAR-RP

Figure 39: Snapshot of synthesized frame - GT_Fly, 0.01bpp.

(a) H.264 (b) HEVC (c) Z-LAR-RP

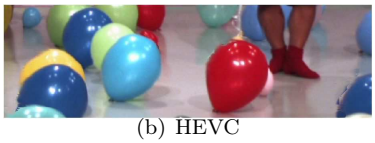
Figure 41: Snapshot of synthesized frame - Newspaper, 0.017bpp.



(a) H.264 (b) HEVC (c) Z-LAR-RP

Figure 42: Snapshot of synthesized frame - Kendo, 0.01bpp.

(a) H.264



(b) HEVC

(c) Z-LAR-RP

Figure 43: Snapshot of synthesized frame - Balloons, 0.01bpp.

7.6 Experiment 2: subjective quality assessment

The experimental protocol presented in this section aims at evaluating the impact of depth-compression-related artifacts on the visual quality of the synthesized views. The subjective image quality evaluation test includes the assessment of state-of-the-art codecs. A first subsection presents the experimental protocol used for assessing the compression methods. A second subsection presents and discusses the results.

7.6.1 Experimental protocol

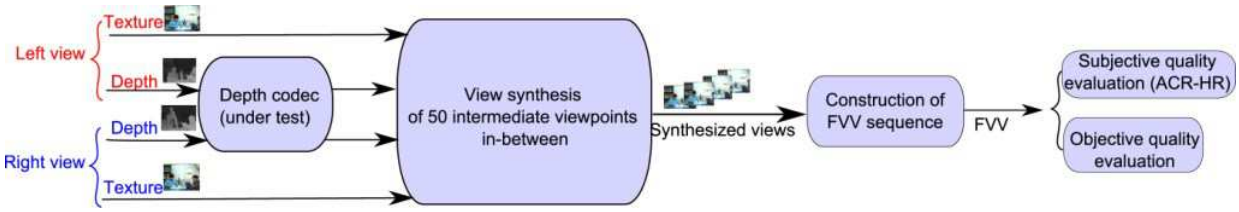


Figure 44: Overview of the experimental protocol.

The goal of this experiment is to determine the performances of the Z-LAR-RP coding method, in terms of subjective quality of the resulting synthesized views. So, we consider the impact of depth compression on the quality of views synthesized from the decoded depth maps quality in a FVV context of use. Only depth maps are

encoded in order to highlight the impact of depth quantization strategies. Fig. 44 depicts the general scheme followed in this experiment. From a given MVD sequence, we consider two different viewpoints and one time t (also referred to as key frames in the following). The associated depth maps are encoded through the depth map codecs under test. From the decoded depth maps, fifty intermediate viewpoints (equally separated) are generated in-between the two considered viewpoints. A sequence of 100 frames (and 10fps) is built from the 50 intermediate virtual frames that simulate a smooth camera motion from left to right and from right to left. This experimental protocol is expected to reveal each coding strategy's distortion specificity. Depth coders under test include the Z-LAR-RP, HEVC 6.1 and H.264 (JM 18), 3D-HTM 0.4 (provided by MPEG) and JPEG2000, all in intra coding mode. For H.264, we used the JM 18.4 (Joint Multiview Video Model) software for the Multiview Video Coding (MVC) project of the Joint Video Team (JVT) of the ISO/IEC Moving Pictures Experts Group (MPEG) [2]. For JPEG2000, a C++ implementation of the JPEG2000 standard was used [3]. In the case of 3D-HTM, inter-view prediction and VSO (View Synthesis Optimization) parameters were enabled. The choice for these methods in this experiment is motivated by the fact that they are reference methods which are usually used as anchors in standardization process. Three test quantization parameters were selected for each depth codec under test according to the visual quality of the rendered views. This procedure was motivated by the need to cover a wide range of categories in the visual quality scale in order to properly define each codec under test. Table 17 gives the details of the quantization parameters used in these experiments. Six MVD sequences are used in these experiments: *Book Arrival*, *Newspaper*, *Kendo* and *Balloons* are real scenes; and *GT_Fly* and *Undo_Dancer* are synthetic scenes. Table 14 summarizes the features of the sequences. The sequences and the key frames were selected for their availability and amount of depth. Table 18 gives the details of the encoded viewpoints and the target viewpoint for the synthesis. The synthesis process is performed through the 3D-HTM 0.4 renderer, that is the view synthesis algorithm used in MPEG 3DV group of standardization at the time of writing this paper. We set the *Blended Mode* parameter of the synthesis algorithm for using the right view only for hole filling instead of carrying out a weigthed average of samples extrapolated from both sides (as done in the MPEG evaluations).

Twenty-seven naive observers participated in the subjective quality evaluation test into two 30-minute sessions. ACR-HR [25] methodology was used to assess 288 FVV sequences, among which were the 96 hereby considered. ACR-HR methodology [25] consists in presenting each stimulus only once to the observers, who are asked to rate the quality of the stimuli relying on a five-level quality scale (5: *Excellent*; 4: *Good*; 3: *Fair*; 2: *Poor*; 1: *Bad*). The reference version of each stimulus is included in the test procedure and rated like any other stimulus. This is reffered to as a

hidden reference condition.The subjective evaluations were conducted in an ITU conforming test environment. The stimuli were displayed on a Panasonic BT-3DL2550 screen (1920×1080p), and according to ITU-T BT.500 [9]. The stimuli sequences with lower resolution (1024x768) were displayed at the sequence resolution with a grey surrounding to fit the Full HD screen.

Depth codec	Quantization parameters
H.264 (JM18)	Qp = [{Book Arrival, Balloons, Kendo, Newspaper}{25, 33, 47}, Undo Dancer{25,40,47}, Gt Fly{30,40,47}]
HEVC 6.1	Qp = [{All of the sequences}{34, 45, 50}]
3D-HTM	Qp = [{All of the sequences}{25, 35, 47}]
JPEG2000	0.05bpp, 0.009bpp and 0.005bpp
Z-LAR-RP	Y = {20, 60, 240 }

Table 17: Quantization parameters used in the experiment.

Sequence Name	Encoded viewpoints	Frame no.
Book Arrival	$10-6$	33
Newspaper	$2-6$	1
Balloons	$1-5$	1
Kendo	$1-5$	1
GT_Fly	$1-9$	157
Undo_Dancer	$1-9$	250

Table 18: Input and output views of the experiment.

7.6.2 Results

From the subjective scores obtained with the ACR-HR method, Mean Opinion Scores (MOS) and Differential Mean Opinion Score (DMOS) are computed between each stimulus and its corresponding (hidden) reference. As recommended in VQEG multimedia Test Plan [59], the DMOS are calculated on a per subject per processed stimulus (PS) basis. The corresponding reference version of the stimulus (SRC) was used to calculate an off-set version of the DMOS value for each PS following the expression:

$$DMOS(PS) = MOS(PS) - MOS(SRC) + 5 \tag{9}$$

In such conditions, the higher the DMOS, the better the quality of the tested stimulus. The lowest bound is 1 as for MOS values but the highest bound can be higher than 5. If the DMOS value is greater than 5, this means that the stimulus is rated better than its corresponding hidden reference. Such values are considered valid by VQEG [59]. Fig. 45 plots the DMOS scores obtained for *Undo Dancer* sequence. In this experimental protocol, the stimuli were not classically selected relying on a list of bit-rates to be evaluated. The stimuli were previously selected by experts based on their subjective visual quality evaluations. For each coding method, the subjective visual quality of the views synthesized from decompressed depth data, at different bit-rates, were first considered by the experts. Then, for each coding method, the experts selected three stimuli corresponding to the categories *Good*, *Fair*, *Poor*. This explains that the obtained curves do not lie in the same bit-rate range. For any coding method, we refer to the highest, the middle and the lowest bit-rates evaluated as $R0$, $R1$ and $R2$

respectively. Fig. 45 shows that in two cases (for Z-LAR-RP and for HEVC coding methods), the observers rated the $R2$ better than $R1$ while the visual quality is expected to fall down when the bit-rate decreases. In the case of Z-LAR-RP, for $R2$, the depth maps used to generate the FVV are almost uniform depth maps. This suggests that a uniform depth map, at low bit-rate, induces less annoying artifacts in the FVV sequence. In the case of HEVC, the depth maps for $R2$ contain smooth edges but the structure of the scene is still perceptible. This suggests that some coding strategies induce coding artifacts whose impact on the visual quality of the synthesized views is reduced and preferable, at low bit-rate.

Fig. 46 shows the DMOS scores obtained for *Balloons* sequence. For three coding

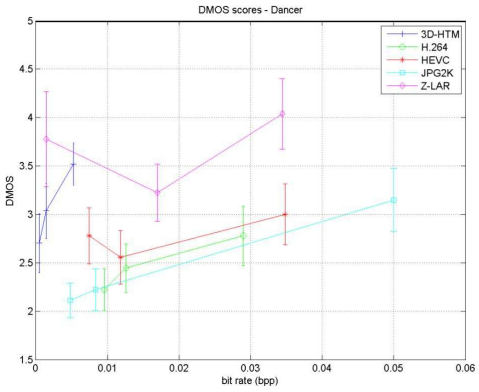


Figure 45: Subjective DMOS over bit-rate - *Undo Dancer*.

methods, DMOS values are higher than 5 (bold black line in the Figure). Since the reference is rated 5 by definition, this means that the processed sequence is rated with a better quality than its associated hidden reference sequence. This can be explained by the fact that depth estimation errors may be smoothed when processed by some compression methods. This is typically the case around object edges, where depth estimation is prone to errors. Some compression methods for some bit-rates may thus smooth inaccurate estimated depth areas, leading to a better visual quality of synthesis. So, we assume that this phenomenon comes from the impact of coding strategies on inaccurately estimated depth maps. This is a particular phenomenon that can be observed in the context of DIBR-synthesized views.

In Fig. 46, the visual quality of $R2$ is also rated better than that of $R1$ with the Z-LAR-RP and the HEVC coding method.

Figures 47, 48 plot the DMOS scores for *Book Arrival* and *Newspaper* (the plots for the other sequences are not presented since the results were similar). In these two figures (47, 48), the Z-LAR-RP coding method also obtains good results in terms of subjective visual quality, at very low bit-rate. These results strengthen the idea that a depth map coding strategy inducing depth fading at low bit rate can enhance the subjective visual quality of the synthesized views. Concerning the performances of the

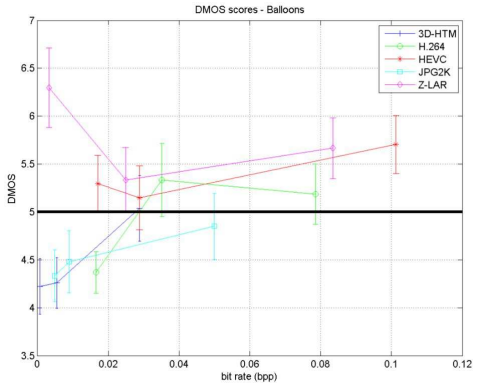


Figure 46: Subjective DMOS over bit-rate - *Balloons*.

compression methods, they seem to vary according to the video content. This is in accordance with the previous comment regarding the impact of the depth estimation accuracy and of the coding strategy on the visual quality of the synthesized views.

3D-HTM includes VSO which modifies the bit-rate distortion trade-off for encoding side depth maps, considering the impact on a synthesized view. The latter is located on the middle view point between the reference view and the current side view. However, FVV requires to synthesize many in-between views with decoded depth optimized for a unique view point. This may explain the low performance on Figures 47 and 48. HEVC outperforms H.264 for all the contents except in the case of *Newspaper*, considering the same range of bit-rate. Similarly, Z-LAR-RP is always rated with the best quality for the considered bit-rate range, except for the cases of *Book Arrival* and *Newspaper*. These examples suggest that a given compression strategy leads to a typical type of distortion that is not perceived or equally accepted depending on the video content. To validate this assumption, an important study on the influence of video contents on compression methods performances is required. We also assume the existence of an impact of MVD sequences features on compression performances.
Finally, an important comment regards the plotted performances of Z-LAR. Except for the cases of *Book Arrival* and *Newspaper*, as previously mentioned, Z-LAR-RP is always rated with the best subjective quality scores. It should be recalled that this compression method relies on a specific strategy which consists in modifying the depth structure of the scene for saving bit-rate. In other words, the lower the bit-rate, the lower the amount of depth in the represented scene. Indeed, in this experiment, the lowest bit-rate corresponds to an almost uniform depth map. And yet, using uniform depth maps for synthesizing new frames amounts to projecting all the reference-colored pixels into the same depth plane. This reduces the errors generally occurring around strong depth discontinuities. Consequently, parallax is significantly reduced in the considered FVV sequences synthesized from these low rate Z-LAR-RP encoded depth maps. For the same reason (uniform depth map), the views rendered from low-bit-

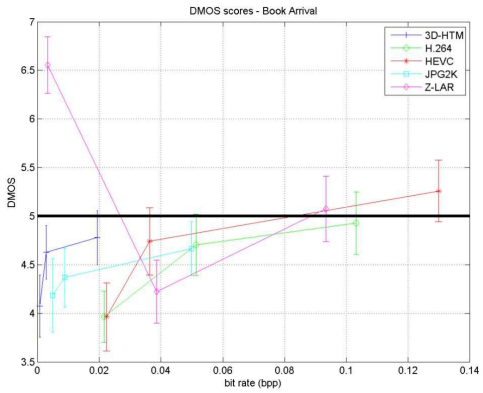


Figure 47: Subjective DMOS over bit-rate - *Book Arrival*.

rate-Z-LAR-RP encoded depth maps are slightly shifted from the targeted virtual viewpoint, as previously observed in Fig. 38, 39, 40, 41, 42 and 43. As a matter of fact, since Z-LAR-RP tends to shift the scene because of the uniform depth maps, the usual full reference quality metrics penalize the method.Yet, the observers rated the subsequent Z-LAR-RP-sequences with the best scores. The observers may have preferred Z-LAR-RP distortions, that is to say, the lack of parallax, over the compression errors that generally appear around object edges as ringing or

crumbling artifacts. However, the observers have rated one factor of the 3D QoE: image quality.

7.7 Conclusion

In this section, we presented a novel approach for depth coding, relying on LAR method. It takes benefit from a pyramidal profile and allows the encoding of multi-resolution depth maps. The enhancement of low resolution depth maps is performed through the help of a region segmentation map obtained from the quad-tree only and improved by the decoded color information. The rate control strategy and the quantization consist in spatially quantizing the depth: the actual depth structure of the scene is modified when the bit-rate decreases, by increasing the homogeneity threshold of the quad-tree partition. The depth map tends to be uniform at very low bit-rates (until 0.003bpp). Altough state-of-the-art coding methods ouperform this novel approach, according to the objective measurements, psycho-visual tests proved that the strategy of Z-LAR-RP enhances the visual quality of the synthesized views, in a FVV context of use. The visual performances achieved thanks to the quantization strategy of Z-LAR-RP show that it may be preferable to transmit less depth depth values than erroneous depth data. The results show that such a depth fading strategy can improve the visual image quality.

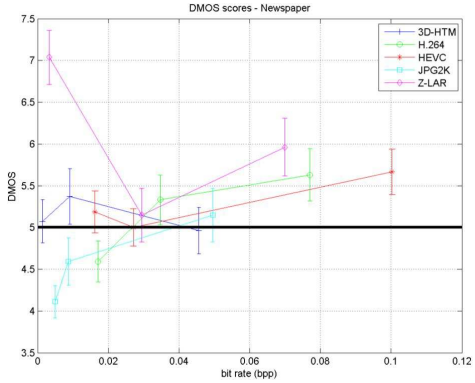


Figure 48: Subjective DMOS over bit-rate - *Newspaper*.

References

[1] "Del 4.2: Representation et codage 3d - rapport intermediaire - definitions des softs et architecture," 2012.

[2] "http://iphome.hhi.de/suehring/tml/," Apr. 2012.

[3] "http://www.kakadusoftware.com/," Apr. 2012.

[4] Z. Belhachmi, D. Bucur, B. Burgeth, and J. Weickert, "How to choose interpolation data in images," *SIAM Journal on Applied Mathematics*, vol. 70, no. 1, pp. 333–352, 2009.

[5] Benjamin Bross and Woo-Jin Han and Jens-Rainer Ohm and Gary J. Sullivan and Ye-Kui Wang and Thomas Wiegand, "High Efficiency Video Coding (HEVC) text specification draft 10 (for FDIS and Consent), Output Document of JCT-VC," 2013.

[6] O. Bici, J. Lainema, and K. Ugur, "Non-CE13: Simplification of merge mode," ITU-T SG16 WP3 & ISO/IEC JTC 1/SC 29/WG 11 JCTVC-G593, November 2011.

[7] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," VCEG-M33, Austin, USA, April 2001.

[8] B. Bross, W.-J. Han, J.-R. Ohm, G. Sullivan, and T. Wiegand, "High Efficiency Video Coding (HEVC) text specification draft 9," ITU-T SG16 WP3 & ISO/IEC JTC 1/SC 29/WG 11 JCTVC-K1003, October 2012.

[9] I. BT., *500, Methodology for the subjective assessment of the quality of television pictures*. November, 1993.

[10] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen, "Unstructured lumigraph rendering," in *Special Interest Group on Computer Graphics and Interactive Techniques (SIGGRAPH), Proceedings of*, New York, NY, USA, 2001, pp. 425–432.

[11] Y.-L. Chang, C.-L. Wu, Y.-P. Tsai, and S. Lei, "CE5.h related: Depth-oriented Neighboring Block Disparity Vector (DoNBDV) with virtual depth retrieval," ITU-T SG 16 WP 3 & ISO/IEC JTC 1/SC 29/WG 11 JCT3V-B0090, October 2012.

[12] Y. Chen, Y.-K. Wang, K. Ugur, M. Hannuksela, J. Lainema, and M. Gabbouj, "The emerging MVC standard for 3D video services," *EURASIP Journal on Advances in Signal Processing*, 2009.

[13] X. Cheng, L. Sun, and S. Yang, "Generation of layered depth images from multi-view video," *IEEE International Conference on Image Processing (ICIP)*, vol. 5, pp. 225–228, Oct. 2007.

[14] G. Cheung, A. Kubota, and A. Ortega, "Sparse representation of depth maps for efficient transform coding," in *IEEE Picture Coding Symposium*, Nagoya, Japan, December 2010.

[15] I. Daribo, G. Cheung, and D. Florencio, "Arithmetic edge coding for arbitrarily shaped subblock motion prediction in depth video compression," in *IEEE International Conference on Image Processing*, Orlando, FL, Sep. 2012.

[16] I. Daribo, C. Tillier, and B. Pesquet-Popescu, "Motion vector sharing and bit-rate allocation for 3D Video-plus-Depth coding," *EURASIP JASP Special Issue on 3DTV*, p. 258920, 2009.

[17] D. De Silva, W. Fernando, H. Kodikaraarachchi, S. Worrall, and A. Kondoz, "A depth map Post-Processing framework for 3D-TV systems based on compression artifact analysis," *IEEE Journal of Selected Topics in Signal Processing*, vol. PP, no. 99, pp. 1–1, 2011.

[18] J. Duan and J. Li, "Compression of the layered depth image," *Image Processing, IEEE Transactions on*, vol. 12, no. 3, pp. 365–372, Mar. 2003.

[19] T. Fujii, K. Mori, K. Takeda, K. Mase, M. Tanimoto, and Y. Suenaga, "Multipoint measuring system for video and sound—100 camera and microphone system," in *IEEE International Conference on Multimedia and Expo*, Toronto, Canada, July 2006.

[20] Gerhard Tech and Krzysztof Wegner and Ying Chen and Sehoon Yea, "3D-HEVC Test Model 2, ISO/IEC JTC1/SC29/WG11," 2012.

[21] S. Gokturk, H. Yalcin, and C. Bamji, "A time-of-flight depth sensor—system description, issues and solutions," in *Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, Washington, DC, June 2004.

[22] D. B. Graziosi, N. M. M. Rodrigues, C. L. Pagliari, S. M. M. de Faria, E. A. B. da Silva, and M. B. De Carvalho, "Compressing depth maps using multiscale recurrent pattern image coding," *Electronics letters*, vol. 46, no. 5, pp. 340–341, 2010.

[23] T. Guionnet, L. Guillo, and C. Guillemot, "CE5.h: Merge candidate list for disparity compensated prediction," ITU-T SG 16 WP 3 & ISO/IEC JTC 1/SC 29/WG 11 JCT3V-B0080, October 2012.

[24] P. Helle, S. Oudin, B. Bross, D. Marpe, M. Bici, K. Ugur, J. Jung, G. Clare, and T. Wiegand, "Block merging for quadtree-based partitioning in HEVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1720 –1731, December 2012.

[25] ITU-T, "Subjective video quality assessment methods for multimedia applications," Geneva, Tech. Rep. Rec. P910, 2008.

[26] W.-S. Kim, A. Ortega, P. Lai, D. Tian, and C. Gomila, "Depth map distortion analysis for view rendering and depth coding," in *IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009.

[27] A. Kubota, A. Smolic, M. Magnor, M. Tanimoto, T. Chen, and C. Zhang, "Multiview imaging and 3DTV," in *IEEE Signal Processing Magazine*, vol. 24, no.6, November 2007.

[28] G. Laroche, J. Jung, and B. Pesquet-Popescu, "RD optimized coding for motion vector predictor selection," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 9, pp. 1247–1257, September 2008.

[29] J.-L. Lin, Y.-W. Chen, Y.-W. Huang, and S. Lei, "3D-CE5.h: Pruning process for the inter-view candidate," ITU-T SG 16 WP 3 & ISO/IEC JTC 1/SC 29/WG 11 JCT3V-B0086, October 2012.

[30] J.-L. Lin, Y.-W. Chen, Y.-P. Tsai, Y.-W. Huang, and S. Lei, "Motion vector coding techniques for HEVC," in *IEEE 13th International Workshop on Multimedia Signal Processing (MMSP)*, October 2011, pp. 1 –6.

[31] Y. Liu, Q. Huang, S. Ma, D. Zhao, and W. Gao, "Joint video/depth rate allocation for 3D video coding based on view synthesis distortion model," *Signal Processing: Image Communication*, vol. 24, no. 8, pp. 666–681, Sep. 2009. [Online]. Available: http://www.sciencedirect.com/science/article/B6V08-4WJHB3J-1/2/df2bb7c1aea8e8c10d3ef1d69f64dd04

[32] K. S. N. F. Y. M. M. Tanimoto, T. Fujii, *Reference Softwares for Depth Estimation and View Synthesis*, ISO/IEC JTC1/SC29/WG11 Std., April 2008.

[33] M. Mainberger and J. Weickert, "Edge-based image compression with homogeneous diffusion," in *Computer Analysis of Images and Patterns*, 2009, pp. 476–483.

[34] M. Maitre, Y. Shinagawa, and M. Do, "Wavelet-based joint estimation and encoding of depth-image-based representations for free-viewpoint rendering," in *IEEE Transactions on Image Processing*, vol. 17, no.6, June 2008, pp. 946–957.

[35] W. Mark, L. McMillan, and G. Bishop, "Post-rendering 3D warping," in *Symposium on Interactive 3D Graphics*, New York, NY, April 1997.

[36] P. Merkle, Y. Morvan, A. Smolic, D. Farin, K. Muller, and T. Wiegand, "The effects of multiview depth video compression on multiview rendering," *Singal Processing: Image Communication*, vol. 24, no. 1-2, pp. 73–88, 2009.

[37] P. Merkle, K. Muller, A. Smolic, and T. Wiegand, "Efficient compression of multiview video exploiting inter-view dependencies based on h. 264/MPEG4-AVC," in *Proc. ICME*, 2006, pp. 9–12.

[38] P. Merkle, A. Smolic, K. Muller, and T. Wiegand, "Efficient prediction structures for multiview video coding," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 17, no. 11, pp. 1461 –1473, nov. 2007.

[39] W. Miled, J. Pesquet, and M. Parent, "A convex optimization approach for depth estimation under illumination variation," *IEEE Transactions on Image Processing*, vol. 18, no. 4, pp. 813–830, April 2009.

[40] E. Mora, J. Jung, B. Pesquet-Popescu, and M. Cagnazzo, "3D-CE5.h related: Modification of the merge candidate list for dependant views in 3DV-HTM," ITU-T SG 16 WP 3 & ISO/IEC JTC 1/SC 29/WG 11 JCT3V-B0069, October 2012.

[41] Y. Morvan, P. de With, and D. Farin, "Platelet-based coding of depth maps for the transmission of multiview images," in *Proceedings of SPIE, Stereoscopic Displays and Applications*, vol. 6055, 2006, pp. 93–100.

[42] Y. Morvan, D. Farin, and P. de With, "Joint depth/texture bit-allocation for multi-view video compression," in *Proceedings of Picture Coding Symposium (PCS 2007)*, vol. 10, Lisboa, Portugal, Nov. 2007, p. 4349.

[43] K. Muller, A. Smolic, K. Dix, P. Merkle, and T. Wiegand, "Coding and intermediate view synthesis of multiview video plus depth," Nov. 2009, pp. 741–744.

[44] F. Pasteau, M. Babel, O. Déforges, C. Strauss, L. Bédat *et al.*, "Locally adaptive resolution (LAR) codec," *Recent Advances in Signal Processing*, pp. 37–48, 2010.

[45] D. Rusanovsky, K. Muller, and A. Vetro, "Common Test Conditions of 3DV Core Experiments," ITU-T SG 16 WP 3 & ISO/IEC JTC 1/SC 29/WG 11 JCT3V-A1100, July 2012.

[46] ——, "Common Test Conditions of 3DV Core Experiments," ITU-T SG 16 WP 3 & ISO/IEC JTC 1/SC 29/WG 11 JCT3V-B1100, October 2012.

[47] M. Sarkis and K. Diepold, "Depth map compression via compressed sensing," in *Proceedings of the 16th IEEE international conference on Image processing*, 2009, pp. 737–740.

[48] D. Scharstein, R. Szeliski, and R. Zabih, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," in *IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV)*, 2001, pp. 131–140.

[49] J. Shade, S. Gortler, L. He, and R. Szeliski, "Layered depth images," in *Computer graphics and interactive techniques, SIGGRAPH.* New York, NY, USA: ACM, 1998, pp. 231–242.

[50] G. Shen, W.-S. Kim, S. Narang, A. Ortega, J. Lee, and H. Wey, "Edge-adaptive transforms for efficient depth map coding," in *IEEE Picture Coding Symposium*, Nagoya, Japan, December 2010.

[51] A. Smolic, K. Müller, N. Stefanoski, J. Ostermann, A. Gotchev, G. Akar, G. Triantafyllidis, and A. Koz, "Coding algorithms for 3dtv - a survey," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 17, no. 11, pp. 1606–1621, Nov. 2007.

[52] A. Srivastava, E. Klassen, S. H. Joshi, and I. H. Jermyn, "Shape analysis of elastic curves in euclidean spaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, pp. 1415–1428, Jul. 2011.

[53] C. Strauss, "Low complexity methods for interpolation and pseudo semantic extraction: applications in the LAR codec," INSA de Rennes, France, Doctoral thesis, 2011.

[54] G. J. Sullivan, P. Topiwala, and A. Luthra, "The h. 264/AVC advanced video coding standard: Overview and introduction to the fidelity range extensions," in *Presented at the SPIE Conference on Applications of Digital Image Processing XXVII Paper No*, vol. 5558, 2004, p. 53.

[55] J. Sung, M. Koo, and S. Yea, "3D-CE5.h: Simplification of disparity vector derivation for HEVC-based 3D video coding," ITU-T SG 16 WP 3 & ISO/IEC JTC 1/SC 29/WG 11 JCT3V-A0126, July 2012.

[56] G. Tech, "HTM-4.1 software," Available: https://hevc.hhi.fraunhofer.de/svn.

[57] G. Tech, K. Wegner, Y. Chen, and S. Yea, "3D-HEVC test model 2," ITU-T SG 16 WP 3 & ISO/IEC JTC 1/SC 29/WG 11 JCT3V-B1005, October 2012.

[58] G. Tech, "HTM-5.0.1 software," Available: https://hevc.hhi.fraunhofer.de/svn/svn_3DVCSoftware/tags/H
5.0.1.

[59] VQEG, "Final report from the video quality experts group on the validation of objective models of multimedia qualiy assessment, phase 1," 2008.

[60] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "The ssim index for image quality assessment," *MATLAB implementation available online from: http://www. cns. nyu. edu/˜ lcv/ssim*, 2003.

[61] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no.7, July 2003, pp. 560–576.

[62] M. Winken, H. Schwarz, and T. Wiegand, "Motion vector inheritance for high efficiency 3D video plus depth coding," in *Picture Coding Symposium (PCS)*, May 2012, pp. 53 –56.

[63] T. Yamamoto and T. Ikai, "Merge candidate refinement for uni-predictive block," ITU-T SG16 WP3 & ISO/IEC JTC 1/SC 29/WG 11 JCTVC-I0293, May 2012.

[64] S.-U. Yoon, E.-K. Lee, S.-Y. Kim, and Y.-S. Ho, "A framework for representation and processing of multi-view video using the concept of layered depth image," *Journal of VLSI Signal Processing Systems for Signal Image and Video Technology*, vol. 46, pp. 87–102, Mar. 2007.

[65] S.-U. Yoon, E.-K. Lee, S.-Y. Kim, Y.-S. Ho, K. Yun, S. Cho, and N. Hur, "Coding of layered depth images representing multiple viewpoint video," *Picture Coding Symposium (PCS)*, vol. SS3-2, pp. 1–6, Apr. 2006.

[66] L. Zhang, Y. Chen, and M. Karczewicz, "3D-CE5.h related: Disparity vector derivation for multiview video and 3DV," ISO/IEC JTC1/SC29/WG11 MPEG2012/m24937, April 2012.

[67] ——, "CE5.h: Disparity vector generation results," ITU-T SG 16 WP 3 & ISO/IEC JTC 1/SC 29/WG 11 JCT3V-A0097, July 2012.

[68] Y. Zhang, G. Y. Jiang, M. Yu, and Y. S. Ho, "Adaptive multiview video coding scheme based on spatiotemporal correlation analyses," *ETRI Journal*, vol. 31, no. 2, April 2009.

[69] C.-L. Zitnick, S.-B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 600–608, 2004.