Immersive Video Coding: Should Geometry Information be Transmitted as Depth Maps?

Patrick Garus, Felix Henry, Joël Jung, Thomas Maugey, Christine Guillemot

Immersive Video Coding: Should Geometry Information be Transmitted as Depth Maps?

Patrick Garus, Felix Henry, *Orange Labs*
Joel Jung, *Tencent Media Lab*
Thomas Maugey, Christine Guillemot, *INRIA*

***Abstract*—Immersive video often refers to multiple views with texture and scene geometry information, from which different viewports can be synthesized on the client side. To design efficient immersive video coding solutions, it is desirable to minimize bitrate, pixel rate and complexity. We investigate whether the classical approach of sending the geometry of a scene as depth maps is appropriate to serve this purpose. Previous work shows that bypassing depth transmission entirely and estimating depth at the client side improves the synthesis performance while saving bitrate and pixel rate. In order to understand if the encoder side depth maps contain information that is beneficial to be transmitted, we first explore a hybrid approach which enables partial depth map transmission using a block-based RD-based decision in the depth coding process. This approach reveals that partial depth map transmission may improve the rendering performance but does not present a good compromise in terms of compression efficiency. This led us to address the remaining drawbacks of decoder side depth estimation: complexity and depth map inaccuracy. We propose a novel system that takes advantage of high quality depth maps at the server side by encoding them into lightweight features that support the depth estimator at the client side. These features allow reducing the amount of data that has to be handled during decoder side depth estimation by 88%, which significantly speeds up the cost computation and the energy minimization of the depth estimator. Furthermore, -46.0% and -37.9% average synthesis BD-Rate gains are achieved compared to the classical approach with depth maps estimated at the encoder.**

***Index Terms*—MPEG, decoder side depth estimation, Feature-Driven Depth Estimation, Immersive Video.**

I. Introduction

THE MPEG-I Visual group is investigating solutions for immersive video to enable 6 Degrees of Freedom (DoF) for the client [1], *i.e.* allowing the user to freely navigate through the video content. Typical applications include sports events, telepresence, entertainment and gaming. For example, in a sport use case, a stadium is surrounded by a camera array that captures the entire field of play so that any point of view can be synthesized. The preferred format adopted by the MPEG-I Visual is multiview plus depth (MVD), where each view carries both a texture and a depth component and the point of view of the user (called the viewport) is synthesized from these components in the client. In order to avoid the deployment of new codecs, the group intends to build its solution around legacy 2D video codecs. Therefore, two solutions are investigated: an MV-HEVC-based [2] and an HEVC-based [3] method. MV-HEVC comes with interview prediction, making it suitable to compress multiview sequences efficiently. However, because of these interview dependencies, it requires the decoding of a large number of views at the client side before the requested viewport can be rendered. This results in pixel processing larger than the pixel rate of the client, *i.e.* the maximum number of pixels it can decode per time unit. While it is possible to discard several views before encoding the quality of the reconstructed scene may be reduced. The other solution is denoted as MPEG Immersive Video (MIV) [4]. While its reference software implementation uses HEVC, it is agnostic to the video codec. Instead of using interview prediction, MIV removes interview-redundancy in a pre-processing step by pruning the input views and packing remaining patches of pixels into atlases. Thus, the pixel rate is significantly reduced. The atlases are subsequently coded using any 2D codec. Yet MIV suffers from similar downsides as the MV-HEVC approach. In both cases, the depth maps are compressed using a video codec designed for texture compression. This affects view synthesis, as the presence of compression artifacts in depth maps impacts the quality of the synthesized views [5]. Furthermore, the depth maps can take up to 30% of the total bitstream and, most significantly, they increase the pixel rate, as they represent additional video frames that need to be decoded. Finally, this method requires the allocation of a suitable quantization parameter (QP) for depth maps, which is not trivial to find [6]. In the following, we denote this kind of system as encoder side depth estimation (ESDE), as the depth maps are acquired or estimated at the encoder side and transmitted to the decoder side.

We have shown in our previous work, that most of the downsides of ESDE are solved by completely bypassing the transmission of depth maps and then performing the depth estimation at the decoder side using the decompressed texture views. This approach is called decoder side depth estimation (DSDE) [7]. In particular, this approach enables a significant improvement in coding efficiency because most of the geometry can be estimated from the decoded textures at the decoder. Furthermore, DSDE reduces the pixel rate as depth maps do not need to be decoded. However, this method requires an additional depth estimation stage in the decoder and may produce depth map inaccuracies [7]. This raises the following question: what part of the geometry should be sent (at the price of a rate cost) and what should be estimated at the decoder (at the price of a possibly degraded view synthesis quality and a higher complexity)? In order to answer this question, we first investigate a compromise between ESDE and DSDE by switching between the two approaches at a Coding Unit (CU) level using a Rate-Distortion criterion. This enables

the transmission of partial depth maps, where only the CUs that are useful to the efficiency of the overall compression efficiency are transmitted. Subsequently, the partial depth is completed on the decoder side using depth estimation. This approach would retain flexibility for content providers, who may still prefer to transmit their own depth maps. It leads to an average BD-Rate reduction of -34.9% and -24.8% for medium and low bitrates respectively, considering synthesized view PSNR compared to the MPEG-I Visual reference. However, it does not substantially outperform regular DSDE. We therefore conclude that it is not beneficial for compression efficiency to send the geometry of a scene as global or local depth maps.
In the second part of this paper, we propose a new design called Feature Driven DSDE, with the goal of solving the two main issues of DSDE: the increase of client side complexity arising from the additional depth estimation step, and the accuracy of the client side depth maps that are estimated from the available decompressed texture views. We use the DSDE approach and, in addition, we allow the transmission of side information in the form of lightweight "features". We extract these features from original depth maps at the block level, in order to guide the decoder side depth estimation. These features include regularization parameters that enable the decoder side depth estimator to generate depth maps which minimize the L2 distance to the original encoder-side depth maps. Furthermore, depth ranges are provided per block, thereby significantly reducing the number of depth hypotheses that need to be evaluated in the decoder. Additional syntax elements serve to refine the system and to reduce the overall bitrate of the features. These features allow the reduction in the number of depth candidates to be explored by 88% on average and by over 95% for several sequences. In addition, the BD-rate gain compared to MPEG-I Visual reference is -46.0% and -37.9% for medium and low bitrates respectively. This approach also outperforms regular DSDE.
The remainder of the paper is organized as follows. In Section II, an overview of possible system architectures for immersive video is provided and the terminology is introduced. Section III presents the Hybrid DSDE and Feature Driven DSDE in more detail. Section IV summarizes the test environment and section V provides the experimental results. Section VI concludes the paper.

II. Background and Related Work

Multiview content can either be based on computer-generated imagery (CGI) or captured by a set of 2D cameras typically positioned in arcs or arrays. The depth maps may either be captured using depth sensors, estimated using depth estimation algorithms or computed from CGI models. Current depth sensors usually do not provide sufficient quality for view synthesis tasks, as they lack both resolution and precision. Natural content therefore relies on depth estimation algorithms. However, choosing the best depth estimator for a view synthesis task is not trivial [8]. For MVD content, intermediate views can be synthesized using depth image-based rendering (DIBR) [9]. Therefore, the support of immersive video implies compression and transmission of multiple viewpoints, view synthesis, and depth estimation.

A. Systems for Immersive Video Coding

The positioning of the depth estimator and the view synthesizer [10] in the codec pipeline defines three possible architectures as illustrated in Fig 1. The scene is captured and stored as multiview textures T. They are compressed using a texture encoder. In case of ESDE, the original depth maps are encoded, typically using the same video codec as for textures. At the decoder side a synthesizer generates novel views S_{ESDE} using decoded textures T^* and depth maps D^* (Fig. 1a). This architecture has been chosen as a reference at the beginning of 3D-HEVC standardization, which has motivated a lot of research in depth coding tools [11], continuously improving compression performance [12] or reducing encoder complexity [13] [14].
Dziembowski et al. [5] have tested the impact of compressing the textures on the accuracy of depth estimation and showed that it is limited to low bitrates. Their investigation is targeting the application of rendering servers, in which depth estimation is performed on compressed textures to save storage, while synthesized views are transferred to the client [15]. This solution is based on the third architecture shown in Fig. 1c), which is significantly different from the design currently considered in MPEG.
We call Basic-DSDE (B-DSDE) the simplest form of DSDE, shown in Fig. 1b). The depth maps are not transmitted, the depth estimator is moved to the decoder side and depth maps D^+ are estimated from decoded textures T^* to synthesize S_{DSDE}. A first analysis of this system compared to ESDE is presented in [7] for full views and in [16] for patch atlases. In contrast to ESDE, little investigation [17] and improvement have been done towards the DSDE system up to this date. In the literature, several reasons are mentioned as for why the DSDE system has not been considered for immersive video [18]:

1) The result of depth estimation varies depending on the method. Consequently, a content provider may not have full control of the view synthesis quality on the consumer's display.
2) Quantization noise and other compression artifacts present in decoded textures make depth estimation more challenging and may harm the view synthesis process.
3) Depth estimation is typically a complex process and may therefore be inappropriate to be performed on the decoder side.

The first argument has only minor relevance because the renderer is purposely not part of the specification of 3D-HEVC and any immersive video standard currently in development: while this prevents a guarantee of a certain level of quality, it leaves room for improvement as any type of future synthesizer can be used. The second argument has been discussed in [7] and we have shown that DSDE provides better view synthesis performance when compared to ESDE if HEVC or MV-HEVC are used for depth compression. In this work, we propose a modification of DSDE that significantly reduces the complexity while further improving the view synthesis quality, thereby addressing the third and last concern related to DSDE.

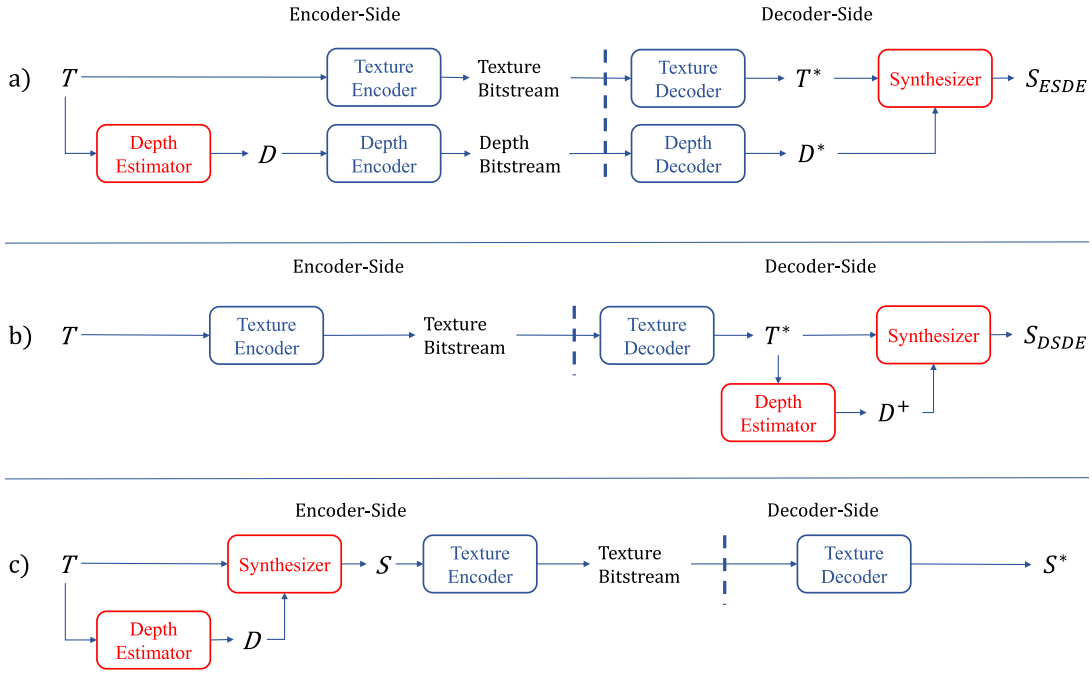

Fig. 1: Three immersive video coding architectures with different positioning of the depth estimator and the view synthesizer. The input signals are original multiview textures T and multiview depth maps D. The superscript $*$ indicates decoded signals. a) shows the ESDE system with synthesized views S_{ESDE}, b) shows the DSDE system with decoder-derived depth D^{+} and synthesized views S_{DSDE} and c) shows a third solution, which directly transmits synthesized views S.

B. Compression for Immersive Video

When compressing MVD video, spatial, temporal, inter-view and inter-component redundancies are exploited to increase the compression performance. MV-HEVC inherently removes redundancy amongst views by using inter-view motion parameters and residual prediction [2]. On one hand, it efficiently exploits previously encoded views for prediction, using only high-level syntax changes compared to HEVC. On the other hand, 3D-HEVC additionally exploits inter-component redundancy between depth maps and textures: block-based synthesized view distortion is used to drive depth map encoding decisions [19], [20]. Substantial research effort has been done towards improving the compression of depth maps with the goal of improving view synthesis [21]. Nevertheless, mostly due to the lack of wide industrial adoption of dedicated depth coding tools, depth maps are compressed using already deployed codecs such as HEVC or MV-HEVC. This comes with a negative impact on the quality of the decoded depth maps for view synthesis. This motivates our work to investigate the estimation of depth maps at the decoder side. As depth estimation and view synthesis are fast advancing topics of research, only the most recent and MPEG-related algorithms are presented in the following.

C. Depth Image-Based Rendering

DIBR systems use depth maps and textures for the rendering of intermediate views. A typical and recent example of such a system is presented in [22], which covers 3D warping, occlusion handling by inpainting and depth map processing. Depth maps are filtered by a local asymmetric depth filter to avoid blurring of non-hole regions. After warping the input views, holes are filled by an additional inverse 3D-Warping step. Finally, a depth map based image inpainting is used to fill remaining occlusions in the synthesized view. In [23], a deep learning based approach is proposed. Depth maps are estimated using two different stereo matching algorithms, one targeting global consistency, the other targetting fine details. Temporal consistency is encouraged in the loss term, which comprises a perceptual loss using activations at different scales of a pretrained VGG16 network. The Versatile View Synthesizer (VVS) is the reference synthesizer for the MV-HEVC-based 6DoF solution in MPEG-I Visual [24]. It was designed for optimizing perceptual quality of virtual views, considering the presence of compression artifacts in the transmitted reference views. Reference views are sorted according to their warping quality, which increases its robustness towards complex camera setups. Depth information is used to project the texture views to the virtual view position. After merging, temporal inpainting is performed to fill remaining holes.

D. Depth Estimation

Like view synthesis, depth estimation is a well studied topic. With the increasing progress of neural networks, most recent approaches use CNN-based methods to estimate high quality depth maps for certain content. Yet, due to their current lack of robustness and their requirement for specific hardware, CNN-based approaches are not yet considered by the standardization community. The authors of [25] use deep learning methods to estimate depth maps from light-fields. Using a fine tuned FlowNet 2.0 network, several candidate depth maps are esti-

mated between the target view and other horizontal or vertical views. They are then merged together to form a single depth map. In addition to depth accuracy, the authors of [26] set the emphasis on reducing the complexity of the depth estimation algorithm by using superpixels as their basic data units. Using a GPU-optimized implementation, they can process one HD depth map per second. The authors of [27] use several runtime optimization strategies on their deep convolutional encoder-decoder design, such as depthwise decomposition, network pruning and hardware-specific compilation. Their CNN-based monocular depth estimator achieves up to 178 fps on 224x224 resolution video.

The Depth Estimation Reference Software [28] is continuously refined in the scope of MPEG-I. DERS8.0 supports up to four neighboring views to estimate a single depth map. Firstly, different depth hypotheses are tested using block matching and the cost for each pixel and depth candidate is computed using a similarity measure. Secondly, the error cost is used to calculate the disparity for each pixel using a graph cut algorithm. The complexity of the block matching stage increases linearly with the number of depth candidates, while the graph cut stage increases polynomially.

In this study, we use DERS8, VVS and MV-HEVC, in MPEG-I Visual testing conditions, in order to show the benefit of estimating depth at the decoder.

III. Investigated DSDE Systems

In this section, we investigate two variants of B-DSDE: one named Hybrid DSDE (H-DSDE) aiming at investigating whether local depth map transmission is efficient, and Feature Driven DSDE, our proposed design to overcome the limitations of B-DSDE: lack of encoder control and high complexity.

A. Hybrid-DSDE

B-DSDE has shown significant quality improvements over ESDE. However, the original depth maps at the encoder side may contain areas which cannot be retrieved at the decoder side with sufficient quality. Therefore it is still possible that *local depth map* transmission is more efficient than B-DSDE, if the encoder has the possibility to carefully select the areas where a depth map is transmitted. To test this hypothesis, we apply a Rate-Distortion (RD) based decision between DSDE and ESDE at the block level (Coding Unit, or CU). This H-DSDE approach aims at transmitting only the depth CUs which are improving the RD criterion. Consequently, there are potentially two different depth estimation processes at the encoder side: a first one to produce the original depth maps (as in ESDE), and a second one which aims at simulating the depth estimation process at the decoder side. In this paper we use DERS8 for both depth estimation processes. The H-DSDE system is shown in Fig. 2a). Texture compression is not modified. We assume textures to be encoded prior to the depth maps. Using the decoded textures, the encoder can estimate the same depth maps as at the decoder side using DERS8.

Inputs to the H-DSDE encoder are the original depth D and DSDE depth D^{+}. First, an ESDE CU is processed by the video codec. The bitrate required to compress the depth CU is denoted as R_{ESDE}. Then, in order to estimate the bitrate required for the DSDE case, an empty (medium grey level) CU is encoded. The goal is to signal the DSDE-decision in the cheapest manner, without disrupting the subsequent compression process. The corresponding bitrate for the DSDE CU is denoted as R_{DSDE}. In order to model the distortion, we follow the concept of the synthesized view distortion change (SVDC) [29] using VVS to synthesize views at source positions. The computation is illustrated in Fig. 4. We construct a complete depth map in order to perform view synthesis. All decisions prior to the current CU being processed are fixed during synthesis and are either ESDE- or DSDE-type depth. All subsequent CUs are replaced by the original depth. Therefore, three depth maps have to be evaluated, in which the current CU is either DSDE, ESDE or original depth. We synthesize a neighboring target view using VVS and compute the sum of squared differences (SSD) between the synthesized view and the corresponding original view T_{Ref} for all three cases. Synthesizing a complete view ensures that the full impact of the CU decision is taken into account in the synthesized view. In practice, the complexity of our encoder could be reduced by synthesizing only the portion of the view that is affected by the CU decision, similar to the partial re-rendering concept of 3D-HEVC [30].

Depending on the depth type, we denote the corresponding distortion measures as e_{ESDE}, e_{DSDE} and e_{orig}. The quantity used in the RD-formulation is the distortion change $\Delta E_{ESDE} = e_{ESDE} - e_{orig}$ and $\Delta E_{DSDE} = e_{DSDE} - e_{orig}$ for ESDE and DSDE accordingly. The decision is taken by choosing the minimum between the cost functions:

$$\begin{aligned} C_{ESDE} &= \Delta E_{ESDE} + \gamma \lambda R_{ESDE}, \\ C_{DSDE} &= \Delta E_{DSDE} + \gamma \lambda R_{DSDE}, \end{aligned} \tag{1}$$

with the lagrangian multiplier λ and a scaling factor γ. γ adapts the lagrangian multiplier in order to optimize the coding performance using the SVDC. We set $\gamma = 0.5$ as in [29]. The decision is signalled through a flag per CU.

At the decoder side, the partial depth map D^{*}_{part} is decoded. A depth estimator reconstructs the depth CUs which were bypassed using the decoded flag. The resulting depth map D^{*+}_{H} is a composition of ESDE and DSDE depth CUs, that can be used to render novel views. Fig. 3 demonstrates the benefit of H-DSDE for the Kitchen sequence. Our H-DSDE encoder is based on HM16.6 [31].

Simulations (see section V) show that H-DSDE improves the PSNR in the majority of sequences. Indeed, the pixel rates reported in Table III indicate that a non negligible amount of depth CUs are selected to be transmitted. Yet, due to the additional bitrate required to transport the local depth maps, it does not outperform B-DSDE in terms of overall compression efficiency (see Table II). This should not be the case in theory, as we use a Rate-Distortion criterion to drive our decisions. However, this criterion relies on approximations, since it uses an existing neighboring original view instead of the desired viewport. This is a limitation inherent to DIBR systems and we believe this is the reason why the encoder fails to select the ESDE blocks most efficiently. Moreover, H-DSDE also negatively impacts the pixel rate since the

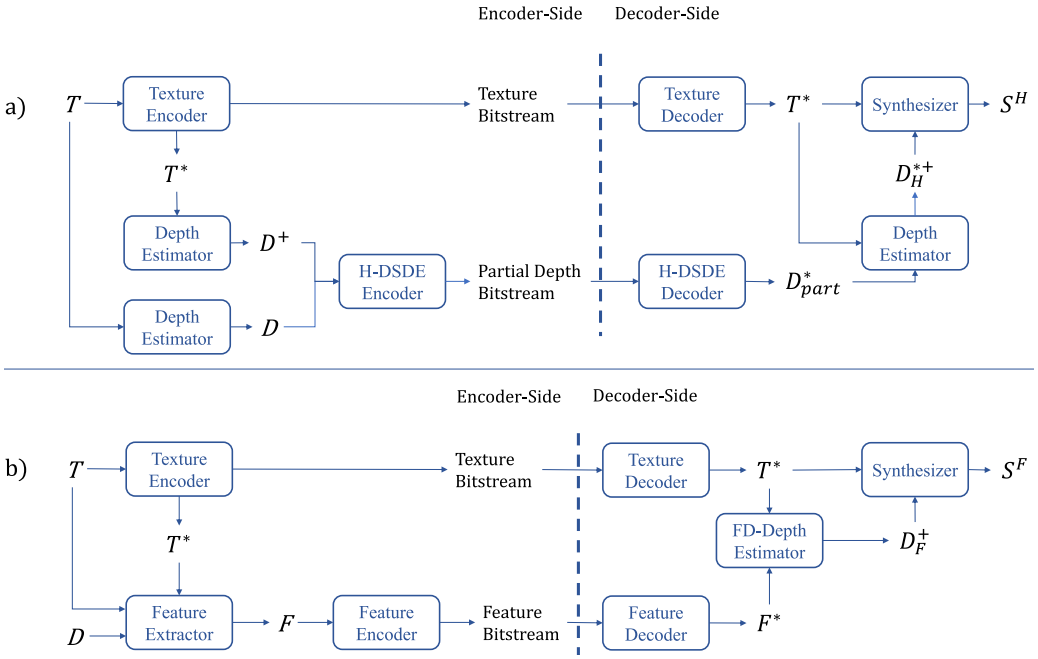

Fig. 2: Overview of the proposed systems. The input signals in both systems are original multiview textures T and multiview depth maps D. T^* are coded textures. a) shows the H-DSDE system. D^+ are depth maps estimated from decoded textures. D^*_{part} are coded partial depth maps and D^{*+}_H are depth maps composed of decoded and decoder derived-depth. S^H are synthesized views. b) shows the FD-DSDE system. F and F^* are the original and coded feature sets respectively. D^+_F are feature-enhanced decoder-derived depth maps. S^F are synthesized views.

Fig. 3: Example for different depth signals in the H-DSDE system. From left to right: source depth (D), DSDE depth (D^+), hybrid depth (D^{*+}_H), decoded partial depth (D^*_{part}).

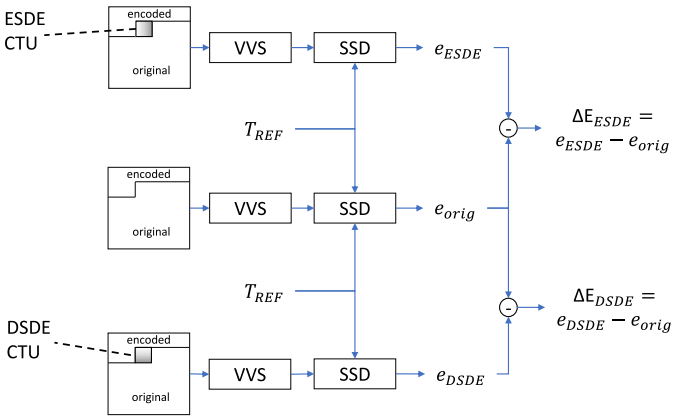

Fig. 4: Overview of the SVDC computation. View synthesis is performed by VVS. The currently encoded CU is either ESDE, DSDE or original. The sum of squard differences (SSD) between the synthesized view and the original view T_{ref} leads to the distortions e_{ESDE}, e_{DSDE} and e_{orig} for all three cases respectively. The differences ΔE_{ESDE} and ΔE_{DSDE} are used in the RD cost computation.

transmitted blocks must be processed by the video decoder in the client. Since the amount of such blocks is unknown, the decoder must have capabilities for the worst case, and therefore have the complexity for pixel rates that are equivalent to the ESDE design. For these reasons, we conclude from the hybrid approach that depth map transmission, even when limited to local cases, does not seem to be a promising design.

B. Feature Driven DSDE

We concluded from our H-DSDE experiments (see section V), that depth map transmission using a video codec, globally or locally, is inefficient. Consequently, we propose a novel system based on B-DSDE, that adresses the two major drawbacks of this design: decoder complexity, and depth estimation inaccuracy. The Feature Driven DSDE (FD-DSDE) system is shown in Fig. 2b). A feature extractor identifies features that help the depth estimation at the decoder side. The features are chosen during encoding time to generate a depth map which minimize the distance to the original depth map. The encoder has access to the decoded version of the textures and can thus simulate the behavior of the decoder side depth estimator in order to correctly choose the feature values. Since the features are extracted and optimized

at the block level, the depth estimator has been modified to operate on a block-basis. For each block, the following features are extracted and transmitted: depth ranges, optimal depth estimation parameters, depth estimation skip flag and a partition flag, which are detailed in the following paragraphs.

Depth Range: traditionally, the camera distance Z_{min} and Z_{max} are required by most depth estimators, in order to define the search range, which can alternatively be described in terms of disparity to a fixed reference view as $d_{max} = \frac{f \cdot b}{Z_{min}}$ and $d_{min} = \frac{f \cdot b}{Z_{max}}$ with the focal length f and the baseline to a reference view b. The disparity range $[d_{min}, ..., d_{max}]$ defines the number of disparity candidates that need to be tested. The number of candidates is $N = (d_{max} - d_{min}) \cdot p + 1$ with optional sub-pixel precision p. The overall size of the cost volume is therefore $CV = N \cdot w \cdot h$ with width w and height h of a view. Each pixel requires an analysis over the full range, which is defined by the farthest and closest object in the view. For each block, the feature extractor simply selects Z_{min} and Z_{max} from the original depth map D. In the same spirit, the available geometry has been utilized previously in order to optimize the computation of plane-sweep volumes in the context of CNN-based view synthesis [32] [33].
Narrowing down depth ranges per block aims at reducing the complexity by minimizing the cost volume that has to be computed. Enabling partial cost volumes increases the speed of block-matching, depth selection (*e.g.* by graph-cuts) and makes the estimated depth maps more accurate by avoiding depth candidates that are out of the correct range. This is illustrated in Fig. 5, where the number of candidates is minimized for smooth areas. The much larger default number of depth candidates is only justified where a block contains a transition between foreground and background.

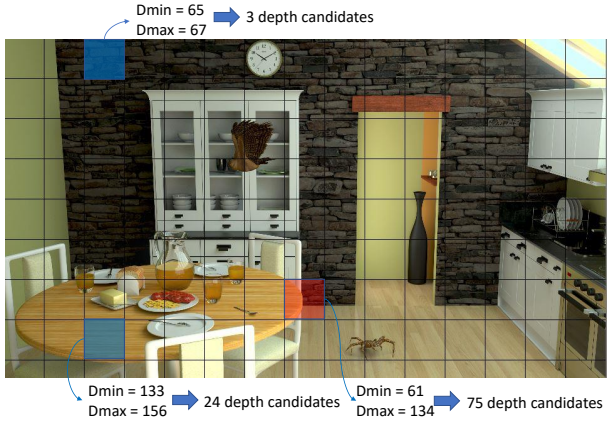

Fig. 5: Illustration of required number of depth candidates per block, specified by the transmitted depth range (D_{min} and D_{max}).

Depth Estimation Skip: the depth estimation skip is signaled through a single flag which indicates if a block requires re-estimation. If a block is skipped, the depth of the previous frame is re-used and the encoder does not transmit any additional features. Fig. 6 indicates, which blocks require re-estimation. In the given example, only around 7.5% of the blocks require a re-estimation of the depth map. The skip

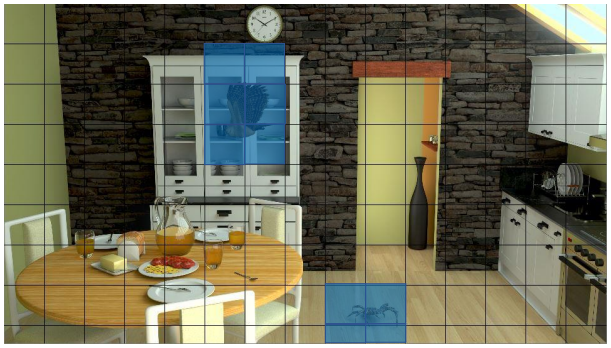

Fig. 6: Illustration of depth estimation skip. Highlighted blocks are re-estimated while all other blocks are skipped.

flag aims at minimizing complexity and maintaining temporal stability in the depth maps and synthesized textures. While it would be possible for the decoder side depth estimator to determine whether a block depth should be updated, this would require additional computation and would go against our objective to reduce the complexity.

Partitioning: with the design of a block-based approach, a maximum block size has to be selected. Partitioning enables the feature extractor to transmit features at a finer granularity if beneficial. We enable a simple partitioning in a quad-tree fashion. The benefit is illustrated in Fig. 7. In the given example, a range of [61,...,134] has to be analyzed in a block of size $N \times N$. The range is large, because the block contains parts of the table (close to camera) and part of the wall behind the chair (far from camera). By partitioning, each block can further minimize the search range, leading to a reduction of depth candidates that have to be tested by an additional 50%.

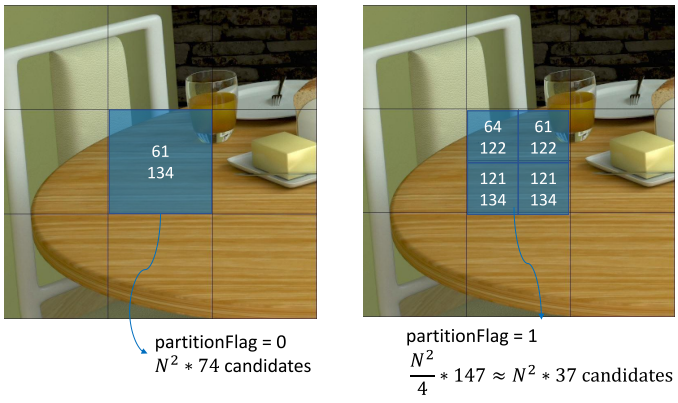

Fig. 7: Illustration of the partition flag. The highlighted block shows D_{min} and D_{max}. On the left, parts of the background are in the block and 74 depth candidates have to be tested. On the right, the partition flag is used and the back- and foreground are more efficiently separated, reducing the total number of depth candidates to 37.

Depth Estimation Parameters: most Multiview or Stereo Depth Estimation methods include a form of regularization which favors homogeneous regions, unless a substantially large change is detected locally. This type of regularization

is usually dependent on parameters determining the amount of change requested to consider a transition as valid, such as smoothing coefficients in energy-minimization methods. We choose to transmit optimized smoothing coefficients as features, because the strength of the regularization is heavily signal-dependent. In our feature-extractor design, several smoothing coefficients are tested by simulating the depth estimation process and selecting the value which minimizes the error with the block in the reference depth.

In this work, our feature extractor selects the features by minimizing the L2 distance of a block between the estimated and the reference depth. In other words, we *encode* the reference depth into a feature set, which helps the depth estimator at the decoder side to estimate depth maps using decoded textures. This criterion is used to identify partitioning and depth estimation parameters. The depth estimation skip is performed if the L2 distance between two temporally adjacent blocks in the original source textures is below a threshold. For CGI content, the depth maps can be used directly to identify the depth estimation skip, where any temporal change implies re-estimation.

Our feature extractor is based on DERS8, where every possible feature set is tested by estimating the depth on a block-basis. The DERS8 used at the decoder side has been modified accordingly, reading the received features and estimating the depth faster and with higher performance. One should note that FD-DSDE does not require any synthesis during encoding, so it is relatively light when compared to other processing stages such as texture coding. The features are compressed using Context-based Binary Arithmetic Coding (CABAC) [34] with a single context assigned for each feature. We encode the features losslessly and therefore optimize for complexity minimization instead of BD-Rate performance, since complexity is the biggest concern of the DSDE system.

Fig. 8 illustrates the impact of our features on depth estimation and synthesis performance. In this example, uncoded textures are used for depth estimation and synthesis. A) shows the performance of unmodified DERS8. F) shows the performance of synthesis if blender depth is used. B)-E) gradually include additional features, extracted from the blender depth. We observe a degradation of quality by moving from a full-frame to a purely block-based method in B). This is expected, as the limited texture available in certain blocks can make the depth estimation more challenging. However, by transmitting the depth range per block in C), many artifacts are removed from the estimated depth map and an improvement of the visual quality is obtained. In D) the depth estimation is enhanced by additional smoothing coefficients, which further reduces the distance to the blender reference and has the biggest impact on the objective quality. In E), partitioning is included, which further refines the depth map locally.

While the difference in objective metrics may seem minor compared to the unmodified DERS8, the impact increases with decoded textures, which we show in section V using DERS8 depth maps as a reference. Additionally, when high-quality depth maps are available (*e.g.* for CGI), this approach can guide the depth estimator to achieve a higher quality depth maps (see section V-D).

IV. Test Environment For Immersive Video

All proposals are evaluated following the Common Test Conditions (CTC) [35] of MPEG-I Visual, detailed in the following.

A. Common Test Conditions

The CTC [35] defines nine test sequences covering a wide spectrum of challenges. The number of cameras, their positioning as well as scene complexity varies for each sequence. A summary of the test sequences is provided in Tab. I. We compute all depth maps for ESDE as well as all DSDE-type proposals using DERS8. Configurations for all sequences are provided by the MPEG-I Visual group, which we adopt for all our experiments without any modification. Since our previous work [7], the Fencing sequence was removed from the mandatory sequences by MPEG, because the unconventional capturing and formatting led to inconclusive results. Therefore, we exclude it from our analysis. Textures and depth maps are compressed independently using MV-HEVC.

Transmitted views reside at source positions. For subjective evaluation, views at intermediate positions are synthesized following a pose trace. For objective evaluation, views at source position are synthesized assuming the texture and depth map at the currently synthesized source position do not exist. In this way, the synthesized view can be directly compared to the uncompressed texture at the source position. View Synthesis is performed by VVS. Textures and depth maps are encoded separately using predefined QPs for textures (QP_T) and depth maps (QP_D). Five QP_T/QP_D-pairs are tested for texture and depth coding. Codecs are compared using the Bjøntegaard delta (BD) [36]. The four upper QP-pairs define the medium-bitrate range and the four lower QP-pairs define the low-bitrate range, which are used for BD-Rate computation. The bitrate consists of all transmitted data which, in our experiments, include textures in all cases. In the cases of ESDE and H-DSDE, the bitrate for encoded depth maps (or partial depth maps) are included as well. In case of FD-DSDE, the rate of the features is included.

In contrast to classical 2D video coding, two types of BD-Rates are reported, which consider the Y-PSNR of either decoded textures (video BD-Rate) or of synthesized textures (synth BD-Rate) together with the total bitrate. Beyond the CTC, MPEG-I Visual further studies the combination of additional metrics with the BD-Rate by replacing the PSNR with Video Multimethod Assessment Fusion (VMAF) [37] and Multi-Scale Structural Similarity (MS-SSIM) [38]. This is motivated by the fact that the evaluation of view synthesis performance can be misleading if merely PSNR is considered [39]. While a common agreement of multiple metrics may confirm an actual improvement, subjective evaluation of synthesized video segments is still the proper way of evaluating changes in the design. We additionally report the LPIPS metric [40], which evaluates the human perception of synthesized views. The overall objective performance of the system is always evaluated considering the bitrate together with quality of synthesized textures compared to the original source textures. According to the conventions of the CTC,

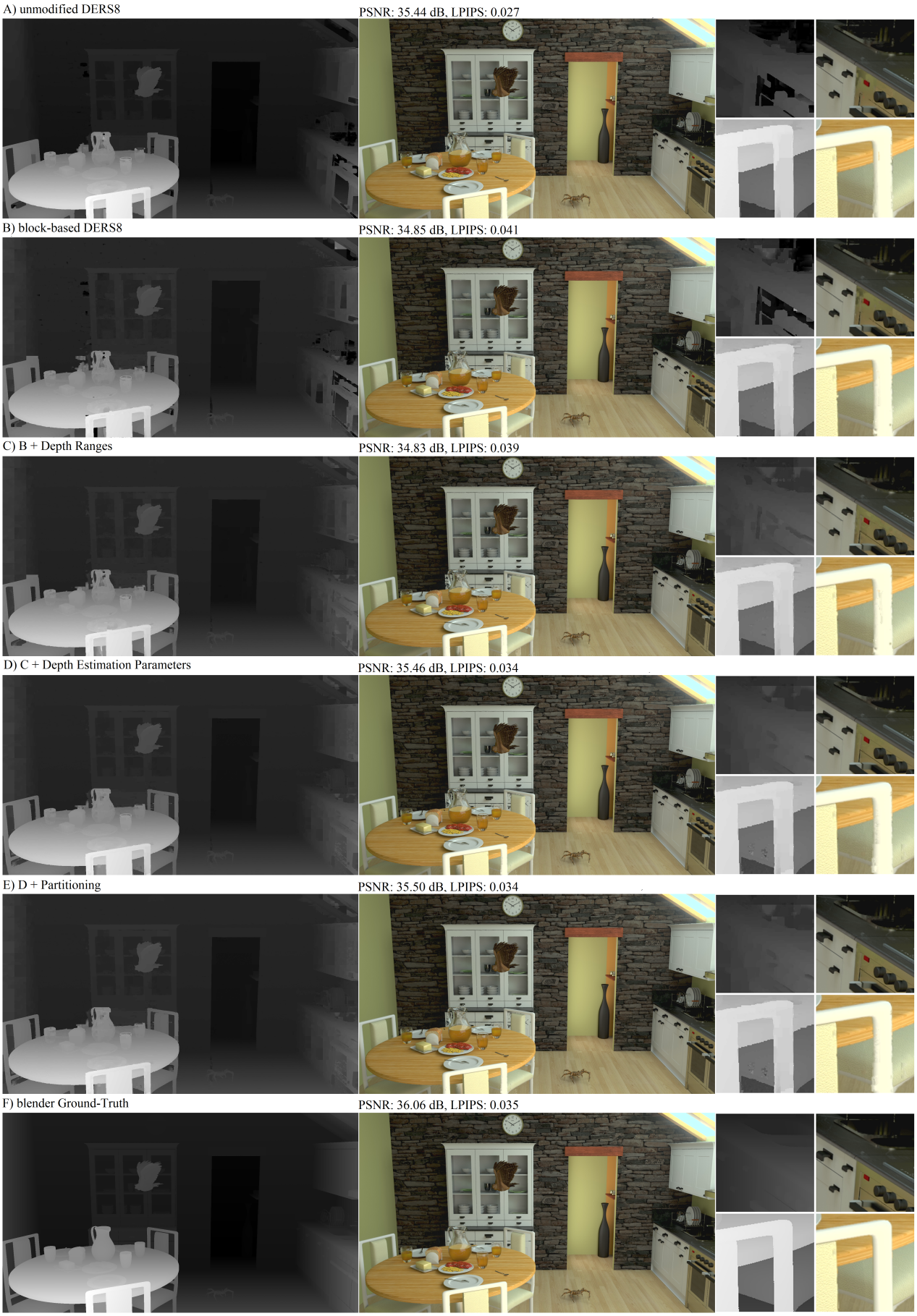

Fig. 8: Illustration of the impact of features, extracted from high quality blender depth. Depth Estimation Parameters and Partitioning are chosen to maximize PSNR.

Sequence	Type	#Views (Setup)	Resolution (frame rate)
Painter	Natural	16 (4x4 planar)	2048x1088 (30)
UnicornA	Natural	25 (5x5 planar)	1920x1080 (still image)
UnicornB	Natural	15 (5x3 planar)	1920x1080 (still image)
Shaman	CGI	25 (5x5 planar)	1920x1080 (30)
Kitchen	CGI	25 (5x5 planar)	1920x1080 (30)
Dancing	CGI	42 (14x3 planar)	1920x1080 (30)
Chef2	Natural	20 (5x4 planar)	1920x1048 (30)
Frog	Natural	15 (15x1 linear)	1920x1080 (30)
Fencing	Natural	10 (10x1 linear)	1920x1080 (25)

TABLE I: Test sequences of the CTC of MPEG-I Visual.

negative BD-Rate values indicate an improvement of the proposal compared to the anchor.

B. Pixel Rate Constraint

Industrial concerns have been raised in MPEG that an immersive video coding standard should reduce the pixel rate to a point where it becomes compatible with current and future devices. Consequently, the group has decided to set two pixel rate constraints [41]. A low pixel rate configuration is constrained to 32 Megapixels at 30 frames per second, motivated by decoding capabilities of modern smartphones. A typical test sequence of size 2048x1088 and 16 views requires 64 Megapixels at 30 frames per second and can therefore not be decoded at the low pixel rate constraints. This motivates solutions which allow transmission of less pixels in terms of texture or depth maps. Consequently, the concept is ordered as follows:

1) First, meet the low pixel rate constraint. Going further down is not considered beneficial at this point.
2) Second, minimize bitrate requirements and maximize synthesis performance, *i.e.* optimize synth BD-Rate.

Due to the increased relevance of pixel rate, an approach that performs worse in terms of BD-Rate may be preferred if it manages to meet the low pixel rate constraint.

C. QP_D Allocation

A common difficulty in the ESDE system is the allocation of QP_D depending on QP_T. Typically, QP_D is derived using

$$QP_D = QP_T + \Delta_{QP}, \tag{2}$$

with a fixed Δ_{QP} for all sequences. The challenge in allocating QP_D arises from its dependency on multiple additional aspects besides the current QP_T: the quality of the given depth maps, the coding method used, and the sequence itself. The allocation problem has been intensively studied in [42]. Recently, the authors of [43] propose to model the relationship between QP_T and QP_D using linear regression, assuming high quality depth maps. Yet, deriving sequence-optimum QP_D remains a challenging task. B-DSDE as well as FD-DSDE do not suffer from this problem: B-DSDE does not require any coding of depth maps while the FD-DSDE proposal encodes the features losslessly.

D. Depth Estimation Complexity

Speed-up techniques such as hardware optimization and parallelization may be used if depth estimation is to be performed at the decoder side. Ideally, depth estimation should only be performed on the views that are required for the synthesis of the viewport. However, there is currently no such depth estimator specifically designed for the DSDE context in MPEG, which could serve as an anchor. Therefore, reporting runtime alone would not fully reflect the potential of the proposals presented in this paper. We chose to additionally report the cost volume reduction (CVR) as this seems to be an accurate approximation of the overall complexity of the depth estimator. This quantity is defined by the following equation:

$$CVR = -\frac{1}{F}\sum_{f=0}^{F-1} 100\frac{refCV - pCV}{refCV}[\%], \tag{3}$$

with $refCV$ being computed using the maximum and minimum disparity of each sequence, while pCV is the partial cost volume, if adapted disparity ranges are transmitted. In a similar manner, pixel rate reduction (PRR) is reported relative to the ESDE anchor (in which all depth maps are transmitted). We use CVR, PRR and runtime to report the impact of our proposed systems on the depth estimation and video decoder complexity respectively.

V. Experimental results

In this section, all results related to B-DSDE, H-DSDE and FD-DSDE are compared to the ESDE anchor of MPEG-I Visual. The objective results are summarized in Tab. II and Tab. III. Examples for view synthesis are shown in Fig. 9.

A. Compression Performance

Depending on the sequence, depth maps can take up to 30% of the total bitrate in ESDE. The video BD-Rate shown in Tab. II reflects the bitrate savings of B-DSDE if depth maps are bypassed entirely. B-DSDE improves ESDE by -17.4% and -21.0% average video BD-Rate for medium and low bitrate respectively. In all our extensions, we decide to reinvest bitrate into either partial depth maps or into features. In case of H-DSDE, transmitting partial depth maps comes with -5.1% and -2.7% average video BD-Rate gain respectively. In case of FD-DSDE, we observe -14.0% and -13.9% video BD-Rate savings for medium and low bitrate respectively. Taking into account the lossless transmission of the features, FD-DSDE proves to be much more efficient than H-DSDE.

B. Synthesis Performance

We reinvest the bitrate into partial depth (H-DSDE) or into features (FD-DSDE) in order to improve the view synthesis quality at the decoder side. Tab. II summarizes all synthesis BD-Rate values with different objective metrics, while Tab. III gives the PSNR and LPIPS values. In case of B-DSDE, the depth estimation at the decoder side leads to an improvement of the objective metrics for all but three cases corresponding to natural sequences. CGI content benefits significantly from the B-DSDE with average synth BD-Rate gains of up to -72.4% (medium) and -51.4% (low) for Dancing. MS-SSIM and VMAF confirm improvements of perceptual quality for all

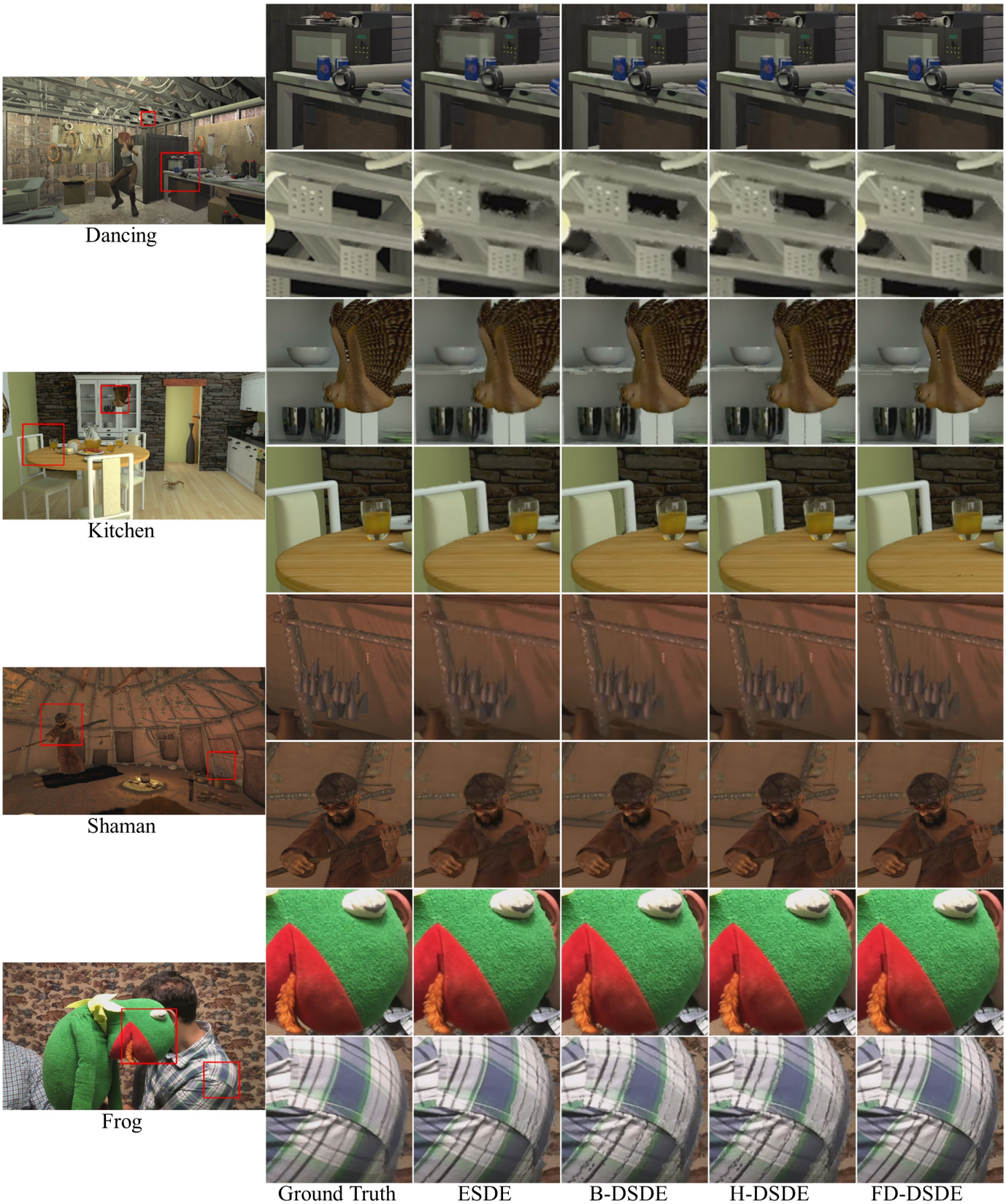

Fig. 9: Ground-truth and synthesized views with the proposed methods, for four sequences: Dancing, Kitchen, Shaman and Frog. FD-DSDE outperforms all other solutions visually.

TABLE II: BD-Rate objective metrics for B-DSDE, H-DSDE and FD-DSDE compared to ESDE. The best synthesis performance is indicated in bold.

Config	Sequence	video BD-Rate [%]			synth PSNR BD-Rate [%]			synth MS-SSIM BD-Rate [%]			synth VMAF BD-Rate [%]		
		B-DSDE	H-DSDE	FD-DSDE	B-DSDE	H-DSDE	FD-DSDE	B-DSDE	H-DSDE	FD-DSDE	B-DSDE	H-DSDE	FD-DSDE
Medium Bitrate	Painter	-36.0	-24.0	-29.3	**-31.7**	-23.0	-25.0	**-30.9**	-19.4	-23.4	**-35.4**	-25.1	-28.6
	UnicornA	-4.7	2.0	-3.8	6.0	-14.6	**-28.5**	-5.9	-4.2	**-13.2**	5.5	-9.8	**-22.4**
	UnicornB	-5.6	-0.8	-4.6	-3.6	-22.7	**-40.4**	-7.8	-7.8	**-14.5**	-4.4	-20.2	**-35.9**
	Shaman	-32.9	-20.9	-28.1	**-55.2**	-46.7	-52.1	-72.4	-31.5	-36.8	**-62.7**	-53.3	-50.8
	Kitchen	-18.2	1.3	-14.8	-39.2	-37.0	**-51.5**	-24.3	-12.4	**-28.6**	-43.3	-34.6	**-52.1**
	Dancing	-5.3	-8.1	-2.7	-72.4	-42.6	**-75.0**	-48.7	-22.8	**-49.0**	**-74.5**	-42.8	-73.8
	Chef2	-27.8	-5.3	-22.8	**-58.4**	-50.9	-42.2	**-40.7**	-25.7	-32.8	**-53.3**	-43.5	-47.5
	Frog	-11.4	-1.3	-5.8	**-57.8**	-41.9	-53.1	**-36.2**	-20.8	-28.6	**-53.3**	-35.7	-48.2
	Average	-17.4	-5.1	-14.0	-39.0	-34.9	**-46.0**	**-33.4**	-18.0	-28.3	-40.8	-33.1	**-44.9**
Low Bitrate	Painter	-39.2	-23.2	-27.6	**-35.1**	-20.8	-23.6	**-34.8**	-18.0	-21.5	**-39.8**	-24.3	-27.7
	UnicornA	-5.4	3.7	-3.4	-6.7	-11.1	**-23.1**	-8.3	-1.1	**-10.6**	-5.0	-7.1	**-18.7**
	UnicornB	-6.7	-0.2	-4.7	-13.8	-15.9	**-26.5**	-9.2	-3.8	**-10.0**	-14.3	-14.7	**-25.2**
	Shaman	-39.7	-20.9	-30.4	**-50.0**	-34.0	-44.8	**-70.1**	-23.0	-33.1	**-60.0**	-42.9	-47.3
	Kitchen	-22.3	6.8	-16.0	-36.0	-21.3	**-43.6**	**-26.0**	0.5	-25.4	-41.8	-21.0	**-45.1**
	Dancing	-8.0	14.7	-2.5	-51.4	-35.1	**-68.0**	-32.9	-11.1	**-40.0**	-59.6	-34.4	**-64.2**
	Chef2	-31.2	-2.2	-22.6	**-54.3**	-36.0	-40.0	**-40.9**	-14.9	-23.4	**-52.8**	-28.8	-36.7
	Frog	-15.0	0.0	-3.7	**-42.8**	-24.8	-34.0	**-28.5**	-9.8	-16.1	**-41.5**	-22.2	-31.3
	Average	-21.0	-2.7	-13.9	-36.3	-24.8	**-37.9**	**-31.3**	-10.2	-22.5	**-39.3**	-24.4	-37.0

TABLE III: Average synthesis PSNR and LPIPS for all systems. The best performance is indicated in bold.

Config	Sequence	synth PSNR [dB]				LPIPS			
		ESDE	B-DSDE	H-DSDE	FD-DSDE	ESDE	B-DSDE	H-DSDE	FD-DSDE
Medium Bitrate	Painter	**34.38**	34.23	34.34	34.25	0.220	0.218	**0.217**	0.218
	UnicornA	29.58	29.38	29.78	**29.92**	0.059	0.063	**0.056**	**0.056**
	UnicornB	29.88	29.84	30.18	**30.42**	0.058	0.059	0.056	**0.055**
	Shaman	33.61	34.21	**34.26**	34.24	0.255	**0.243**	**0.243**	0.250
	Kitchen	30.55	31.05	31.41	**31.45**	0.181	0.176	0.170	**0.169**
	Dancing	28.15	29.97	29.08	**30.06**	0.218	**0.191**	0.202	**0.191**
	Chef2	31.50	31.91	**32.03**	31.70	0.260	0.260	0.258	**0.257**
	Frog	26.95	**27.59**	27.41	27.53	0.229	**0.208**	0.215	0.215
	Average	30.57	31.02	31.06	**31.20**	0.185	0.177	0.177	**0.176**
Low Bitrate	Painter	**32.96**	32.75	32.87	32.80	0.334	**0.331**	**0.331**	0.332
	UnicornA	28.21	28.20	28.54	**28.67**	0.123	0.122	0.116	**0.115**
	UnicornB	28.56	28.65	28.90	**29.12**	0.123	0.121	0.119	**0.118**
	Shaman	32.46	32.75	32.87	**32.93**	0.430	**0.418**	0.419	0.423
	Kitchen	29.42	29.79	30.17	**30.27**	0.324	0.317	0.311	**0.308**
	Dancing	27.02	28.11	27.93	**28.88**	0.386	0.374	0.370	**0.356**
	Chef2	30.82	31.29	**31.41**	31.13	0.323	0.322	0.321	**0.319**
	Frog	25.99	26.59	26.44	**26.60**	0.363	**0.337**	0.344	0.346
	Average	29.43	29.77	29.89	**30.05**	0.301	0.293	0.291	**0.290**

sequences, including those that suffer in terms of PSNR-based synthesis BD-Rate.

H-DSDE improves the PSNR performance for the majority of sequences beyond B-DSDE by transmitting the relevant depth map CTUs which cannot be estimated accurately at the decoder side. Overall, the PSNR improves on average by 0.55 dB and 0.34 dB for medium and low bitrates respectively. As expected, the improvement is better at low bitrates, since B-DSDE is more challenging with highly compressed textures. An exception is Dancing, which performs worse in H-DSDE than in B-DSDE. This is likely due to imperfection in the reference view selection during H-DSDE encoding: we select between ESDE and DSDE by synthesizing a single target view. In the final evaluation however, the depth map is used for the synthesis of the neighborhood, which can be up to four target views at source position. Having the most numbers of views, this disadvantage becomes visible in the Dancing sequence. Accordingly, the synth BD-Rate for H-DSDE improves by -34.9% and -24.8% for medium and low bitrate respectively. While the perceptual quality is still improved according to the MS-SSIM metric, it cannot outperform B-DSDE. The VMAF metric confirms a lack of perceptual quality in the H-DSDE solution, which may be due to the composition of depth maps from two sources. Discontinuities at block-boundaries can translate into visually unpleasant artifacts. In conclusion, we are able to identify relevant ESDE-CUs, which improve the synthesis PSNR compared to B-DSDE. However, the depth map compression performance is too low to transport this relevant information to the decoder side. The objective quality with FD-DSDE is improved further compared with B-DSDE and H-DSDE. PSNR gains increase on average by 0.63 dB and 0.62 dB for medium and low bitrate respectively. Similar to H-DSDE, the benefit of the method heavily depends on the quality of the depth map at the encoder side. In the case of the Chef2 sequence, we observe that neither of our extensions led to a benefit over B-DSDE, which indicates that the encoder side depth maps did not contain relevant additional information compared to the B-DSDE depth maps. However, the Dancing sequence increases in PSNR by 1.91 dB, surpassing B-DSDE and H-DSDE. Overall, we observe an average synth BD-Rate reduction of 46% and 37.9% showing a clear benefit over both B-DSDE and H-DSDE. In contrast to ESDE and H-DSDE, FD-DSDE efficiently transmits the relevant depth information in terms of features to the client side. The MS-SSIM and VMAF metrics confirm significant improvements compared to ESDE, with equivalent performance compared to B-DSDE.

For the majority of sequences, the LPIPS metric is better with FD-DSDE, showing that the synthesis quality is improved. In Fig. 9 all methods are compared visually. Double-contouring is a typical synthesis artifact in the case of ESDE. In B-DSDE we observe much less blurring but more noise-like distortions. H-DSDE can solve some of these artifacts, if the RD compromise is reasonable. Therefore, H-DSDE may be an improvement overall, but we still find the same artifacts as in B-DSDE or ESDE in some areas. FD-DSDE does not contain any ESDE-related artifacts. Instead, it is a direct improvement of B-DSDE. Most of the noise-like artifacts are significantly reduced, as is visible in the Frog sequence. Most edges appear much cleaner and the views are better aligned in the blending process as seen by the Dancing example.

C. Complexity Reduction

Pixel Rate Reduction (PRR) and Cost Volume Reduction (CVR) are summarized in Tab. V. In H-DSDE, the pixel rate varies depending on the amount of selected ESDE CUs. The total cost volume increases as more DSDE CUs are selected. Additionally, the PRR reflects the amount of DSDE or ESDE CUs selected in the process. While true PRR would require to pack the ESDE-blocks into a patch atlas, we can still predict the potential benefit in H-DSDE.

On average, we could save around 28.68% and 30.98% pixel rate in case of H-DSDE for medium and low bitrates respectively. In other words, around 20% of the pixel rate has been reinvested into depth map transmission in contrast to B-DSDE. Whenever an ESDE CU has been transmitted, the decoder does not need to estimate the corresponding depth. We interpret this as a cost volume of size zero, which allows us to estimate the total CVR in case of H-DSDE. On average, we reduce the cost volume size by -38.49% and -36.11%. In the case of FD-DSDE, which does not entail any depth map transmission, we maintain the -50% pixel rate reduction of B-DSDE. This is because the defined features are not appropriate to be transmitted using a video encoder and are instead contained in the metadata of an immersive video bitstream. Independent of the pixel rate, the CVR of FD-DSDE is 88.3% and 88.4% for medium and low bitrate respectively.

The depth estimation is computationally the most demanding stage in the client and requires more processing time than video decoding, view synthesis and feature decoding. Therefore, the CVR is expected to have a very significant impact on the total amount of processing required in the client. The software implementation has not been optimized, however we show in Tab IV the runtime values of our fastest proposal FD-DSDE compared to the unmodified DERS8. Computation was performed on an Intel Xeon (R) CPU E5-2520 0 @ 2.00 GHz. On average, we measure a speedup of 18.5. The speedup depends on the cost volume reduction and the amount of motion in the sequences.

D. FD-DSDE with High Quality Depth Maps

Up to this point, all depth maps used in our experiments originated from the same algorithm (DERS8). In this way we show that the changes in performance of all proposed systems do not come from changes in the depth estimation algorithm. Yet, in practice, one would use the best depth maps that are available. Due to the high potential of FD-DSDE, shown in the last sections, we evaluate it once again using high quality blender depth for CGI content. Tab. VI shows the results of all objective metrics, where the Kitchen, Shaman and Dancing sequences have been re-computed using high-quality depth maps as a reference. Synth BD-Rate is further improved compared to the FD-DSDE results with DERS8-reference. The PSNR of the Dancing sequence has improved by 2.86 dB on average, which is 1 dB more than B-DSDE and 2 dB more than H-DSDE, while simultaneously keeping the bitrate low. This further underlines that FD-DSDE is a more efficient way of taking advantage from high quality depth maps by encoding the relevant geometry information into features.

E. Comparison with 3D-HEVC

The previous section reports experimental results using MPEG-I Visual CTC. As explained above, MPEG-I Visual chose to facilitate the adoption of its standard by considering existing 2D video codec for depth and textures, thus preventing the use of specific depth coding and interview prediction tools. In this subsection, we relax this constraint, and test our FD-DSDE approach against the 3D-HEVC standard. As there are inherent limitations of the 3D-HEVC standard and its reference software that prevent us from setting it into a configuration that matches the MPEG-I Visual CTC, we chose to use the 3D-HEVC test conditions as described in [44] and

TABLE IV: Average runtime per frame and view using the unmodified DERS8 and FD-DSDE.

Sequence	DERS8 [s]	FD-DSDE [s]	speedup
Painter	698.3	111.0	6.3
UnicornA	291.4	40.4	7.2
UnicornB	270.5	35.2	7.7
Shaman	3933.9	169.1	23.3
Kitchen	1051.5	22.6	46.5
Dancing	1195.0	30.2	39.6
Chef2	2565.1	228.8	11.2
Frog	1107.3	177.6	6.2
Average	**1389.1**	**101.85**	**18.5**

TABLE V: Pixel Rate and Cost Volume Reduction for H-DSDE and FD-DSDE.

Config	Sequence	PRR [%]		CVR [%]	
		H-DSDE	FD-DSDE	H-DSDE	FD-DSDE
Medium Bitrate	Painter	-39.43	-50.0	-21.13	-83.6
	UnicornA	-22.55	-50.0	-54.89	-76.8
	UnicornB	-21.13	-50.0	-28.83	-84.0
	Shaman	-41.05	-50.0	-17.89	-94.8
	Kitchen	-32.24	-50.0	-35.51	-97.7
	Dancing	-14.78	-50.0	-70.42	-97.3
	Chef2	-30.64	-50.0	-34.58	-88.8
	Frog	-27.65	-50.0	-44.68	-82.4
	Average	**-28.68**	**-50.0**	**-38.49**	**-88.3**
Low Bitrate	Painter	-39.78	-50.0	-20.42	-83.8
	UnicornA	-19.30	-50.0	-61.39	-77.4
	UnicornB	-32.16	-50.0	-35.7	-84.4
	Shaman	-45.13	-50.0	-9.72	-95.0
	Kitchen	-35.60	-50.0	-28.79	-97.9
	Dancing	-18.46	-50.0	-63.07	-97.3
	Chef2	-30.64	-50.0	-20.42	-88.7
	Frog	-26.80	-50.0	-46.39	-82.4
	Average	**-30.98**	**-50.0**	**-36.11**	**-88.4**

TABLE VI: Objective metrics, where features are extracted from high quality depth maps created with blender, which are available for the CGI sequences (bold). The value in brackets indicate the change to the results of Tab. II and III.

Config	Reference Depth	Sequence	video BD-Rate [%]	synth PSNR BD-Rate [%]	synth MS-SSIM BD-Rate [%]	synth VMAF BD-Rate [%]	synth PSNR [dB]	synth LPIPS
Medium Bitrate	DERS8	Painter	-29.3	-25.0	-23.4	-28.6	34.25	0.218
		UnicornA	-3.8	-28.5	-13.2	-22.4	29.92	0.054
		UnicornB	-4.7	-40.4	-14.5	-35.9	30.42	0.055
		Chef2	-22.8	-42.2	-32.8	-47.5	31.70	0.256
		Frog	-5.8	-53.1	-28.6	-48.2	27.53	0.216
	Blender	**Shaman**	**-22.0 (+6.1)**	**-58.3 (-6.2)**	**-40.6 (-3.2)**	**-48.9 (+1.9)**	**34.91 (+0.67)**	**0.243 (-0.007)**
		Kitchen	**-15.2 (-0.4)**	**-56.7 (-5.2)**	**-32.6 (-4.0)**	**-53.7 (-1.6)**	**31.74 (+0.29)**	**0.166 (-0.003)**
		Dancing	**-3.4 (-0.7)**	**-81.4 (-6.4)**	**-56.6 (-7.6)**	**-82.4 (-8.6)**	**31.03 (+0.97)**	**0.172 (-0.018)**
		Average	**-13.4**	**-48.2**	**-30.3**	**-45.9**	**31.44**	**0.173**
Low Bitrate	DERS8	Painter	-27.6	-23.6	-21.5	-27.7	32.80	0.332
		UnicornA	-3.4	-23.1	-10.6	-18.7	28.67	0.114
		UnicornB	-4.7	-26.5	-10.0	-25.2	29.12	0.118
		Chef2	-22.6	-40.0	-23.4	-36.7	31.13	0.319
		Frog	-3.7	-34.0	-16.1	-31.3	26.60	0.345
	Blender	**Shaman**	**-18.7 (+11.7)**	**-40.1 (+4.7)**	**-24.5 (+8.6)**	**-36.3 (+11.0)**	**33.36 (+0.43)**	**0.423 (-0.000)**
		Kitchen	**-16.8 (-0.8)**	**-47.7 (-4.1)**	**-28.2 (-2.8)**	**-46.4 (-1.3)**	**30.46 (+0.19)**	**0.306 (-0.002)**
		Dancing	**-4.1 (-1.6)**	**-73.6 (-5.6)**	**-46.2 (-6.2)**	**-72.0 (-7.8)**	**29.51 (+0.63)**	**0.346 (-0.010)**
		Average	**-11.9**	**-38.6**	**-22.6**	**-36.8**	**30.21**	**0.288**

TABLE VII: Synthesis PSNR BD-Rates using the 3-View configuration of the 3D-HEVC CTC. FD-DSDE average (*) is underestimated due to missing overlaps for Frog and Dancing. Instead, their RD-curves are shown in Fig. 10.

Config	Sequence	synth PSNR BD-Rate [%]		
		3D-HEVC	B-DSDE	FD-DSDE
Medium Bitrate	Painter	-24.3	**-36.4**	-27.7
	UnicornA	-58.3	-42.3	**-64.6**
	UnicornB	**-59.2**	-33.8	-44.1
	Shaman	**-59.5**	-46.8	-35.9
	Kitchen	**-66.4**	-61.1	-57.2
	Dancing	-62.2	-83.4	**no overlap**
	Chef2	-67.1	**-82.3**	-47.9
	Frog	-62.8	-79.9	**no overlap**
	Average	-57.5	-58.2	-46.0*
Low Bitrate	Painter	-25.0	**-36.5**	-27.0
	UnicornA	-39.3	-38.3	**-47.5**
	UnicornB	**-41.6**	-37.4	-40.0
	Shaman	**-50.6**	-25.5	-22.7
	Kitchen	**-57.4**	-50.2	-53.6
	Dancing	-32.8	-82.2	**no overlap**
	Chef2	-55.3	**-61.5**	-40.7
	Frog	-40.5	-58.6	**-77.8**
	Average	-42.8	-48.8	-44.2*

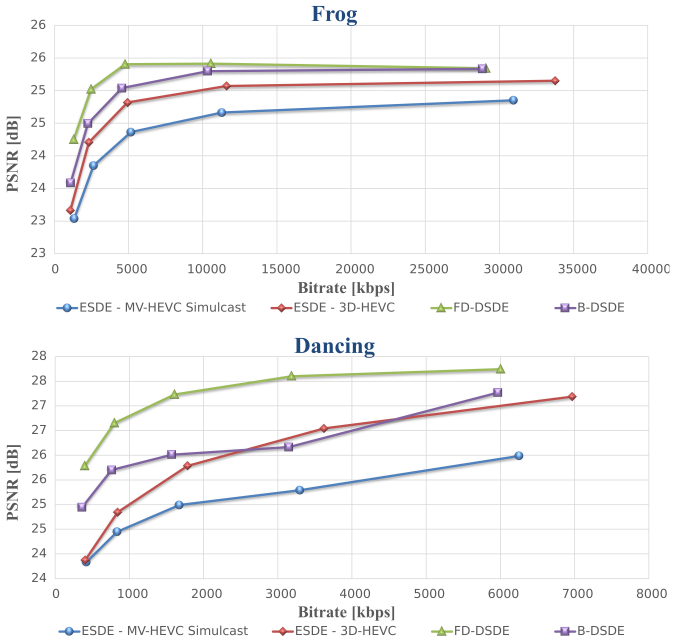

Fig. 10: RD-curves for the Dancing and Frog sequences.

set the configuration of FD-DSDE accordingly: in particular, we limit ourselves to a 3 view configuration, using the first three horizontal views of the MPEG-I Visual test sequences. The results are summarized in Tab. VII and Fig. 10. To be consistent with our previous section, the MV-HEVC anchor in 3-view configuration is set as an anchor. As expected, 3D-HEVC outperforms MVD compression with MV-HEVC simulcast. In many cases B-DSDE can still outperform 3D-HEVC. Since depth maps are estimated from 3 views only, FD-DSDE may perform worse, as the features are extracted from depth maps with more artifacts. We show the RD curves in Fig. 10.

VI. Conclusion

The main conclusion we draw from our DSDE experiments is that depth maps are mostly redundant, as geometry is already transported within the textures. We are able to extract them at the decoder side with enough accuracy to efficiently drive the synthesis. This was already shown in earlier B-DSDE experiments. While B-DSDE performs well, we have explored the hypothesis that it could still be useful to transmit some parts of the depth map in our H-DSDE experiments, where blocks of the depth maps are omitted or transmitted based on an encoder criterion. Synthesis PSNR shows that no significant gain over B-DSDE can be obtained. Also, the hybrid approach still requires the transmission of partial depth maps, which negatively impacts the pixel rate. Therefore, this suggests that depth map transmission using a generic video codec is not the most promising approach. This motivated our final proposal of FD-DSDE, which transmits intuitive and versatile features such as depth ranges and regularization strengths. Furthermore, the block-based depth estimation approach is amenable to parallelization. It is asserted that these design choices can lead to a significant reduction of complexity,

making decoder side depth estimation much more attractive. The authors plan to explore in the future other types of features to help the synthesis.

ACKNOWLEDGMENT

The authors would like to thank Gordon Clare, from Orange Labs, for providing an implementation of CABAC.

REFERENCES

[1] M. Wien, J.M. Boyce, T. Stockhammer, W.-H. Peng, "Standardization Status of Immersive Video Coding," IEEE J. on Emerging and Selected Topics in Circuits and Systems, vol. 9, no. 1, pp. 5-17, Mar. 2019.

[2] G. Tech, Y. Chen, K. Müller, J.-R. Ohm, A. Vetro, Y.-K. Wang, "Overview of the Multiview and 3D Extensions of High Efficiency Video Coding," IEEE Trans. on Circuits and Systems for Video Technology, Jan. 2019;26(1):35-49.

[3] G. J. Sullivan, J. Ohm, W. Han and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," IEEE Trans. on Circuits and Systems for Video Technology, vol. 22, no. 12, pp. 1649-1668, Dec. 2012.

[4] J. M. Boyce, R. Doré, A. Dziembowski, J. Fleureau, J. Jung, B. Kroon, B. Salahieh, V. K. M. Vadakital and L. Yu, "MPEG Immersive Video Coding Standard," Proc. of the IEEE.

[5] A. Dziembowski, M. Domański, A. Grzelka, D. Mieloch, J. Stankowski and K. Wegner, "The influence of a lossy compression on the quality of estimated depth maps," Int. Conf. on Systems, Signals and Image Process. (IWSSIP), pp. 1-4, 2016.

[6] E. Bosc, V. Jantet, M. Pressigout, L. Morin and C. Guillemot, "Bit-rate allocation for multi-view video plus depth," 2011 3DTV Conf.: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON), Antalya, pp. 1-4, 2011.

[7] P. Garus, J. Jung, T. Maugey and C. Guillemot, "Bypassing Depth Maps Transmission For Immersive Video Coding," Picture Coding Symposium (PCS), pp. 1-5, 2019.

[8] A. Q. de Oliveira, T. L. T. da Silveira, M. Walter, C.R. Jung, "On the Performance of DIBR Methods When Using Depth Maps from State-of-the-art Stereo Matching Algorithms," IEEE Int. Conf. on Acoustics, Speech and Signal Processing, pp. 2272-2276, 2019.

[9] C. Fehn, "Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV," Proc. Electron. Imag., pp. 93–104, 2004.

[10] M. Tanimoto, "FTV (free viewpoint television) creating ray-based image engineering," IEEE Int. Conf. on Image Process. pp. II-25, 2005.

[11] G. Sanchez, L. Agostini, C. Marcon, "Algorithms for Efficient and Fast 3D-HEVC Depth Map Encoding," Springer Nature, 2019.

[12] O. Stankiewicz, K. Wegner and M. Domański, "Study of 3D Video Compression Using Nonlinear Depth Representation," IEEE Access, vol. 7, pp. 31110-31122, 2019,

[13] J. Lei, J. Duan, F. Wu, N. Ling and C. Hou, "Fast Mode Decision Based on Grayscale Similarity and Inter-View Correlation for Depth Map Coding in 3D-HEVC," IEEE Trans. on Circuits and Systems for Video Technology, vol. 28, no. 3, pp. 706-718, Mar. 2018.

[14] M. Saldanha, G. Sanchez, C. Marcon and L. Agostini, "Fast 3D-HEVC Depth Map Encoding Using Machine Learning," IEEE Trans. on Circuits and Systems for Video Technology, vol. 30, no. 3, pp. 850-861, Mar. 2020.

[15] O. Stankiewicz, M. Domański, A. Dziembowski, A. Grzelka, D. Mieloch and J. Samelak, "A Free-Viewpoint Television System for Horizontal Virtual Navigation," IEEE Trans. on Multimedia, vol. 20, no. 8, pp. 2182-2195, Aug. 2018.

[16] M. Milovanović, F. Henry, M. Cagnazzo and J. Jung, "Patch Decoder-Side Depth Estimation in MPEG Immersive Video," IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP), pp. 1945-1949, 2021.

[17] C. Brites, J. Ascenso and F. Pereira, "Epipolar plane image based rendering for 3D video coding," IEEE Int. Workshop on Multimedia Signal Process. (MMSP), pp. 1-6, 2015.

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

[19] W. Kim, A. Ortega, P. Lai, D. Tian, "Depth Map Coding Optimization Using Rendered View Distortion for 3D Video Coding," IEEE Trans. on Image Process., vol. 24, no. 11, pp. 3534-3545, Nov. 2015.

[20] B. Rajei, T. Maugey, P. Frossard, "Rate-distortion analysis of multiview coding in a DIBR framework," annals of telecommunications, Nov. 2012.

[21] C. Debono, M. Domański, S. De Faria, K. Klimaszewski, L. Lucas, N. Rodrigues, K. Wegner, "Efficient Depth-Based Coding," in 3D Visual Content Creation, Coding and Delivery, pp.97-114, Jan. 2019.

[22] S. Zhu, H. Xu and L. Yan, "An Improved Depth Image Based Virtual View Synthesis Method for Interactive 3D Video," IEEE Access, vol. 7, pp. 115171-115180, 2019,

[23] P. Hedman, J. Philip, T. Price, J.-M. Frahm, G. Drettakis, G. Brostowruit, "Deep Blending for Free-Viewpoint Image-Based Rendering," ACM Trans. Graph. 37, 257:1-257:15, 2018.

[24] P. Boissonade and J. Jung, "[MPEG-I Visual] Improvement of VVS1.0.1," ISO/IEC JTC1/SC29/WG11/MPEG2019/m46263, Jan. 2019.

[25] J. Shi, X. Jiang, C. Guillemot, "A framework for learning depth from a flexible subset of dense and sparse light field views," IEEE Int. Conf on Acoustics, Speech, and Signal Process., 13-17 May 2019.

[26] A. Chuchvara, A. Barsi, A. Gotchev, "Fast and Accurate Depth Estimation from Sparse Light Fields," arXiv:1812.06856, Dec. 2018.

[27] D. Wofk, F. Ma, T.-J. Yang, S. Karaman, V. Sze, "FastDepth: Fast Monocular Depth Estimation on Embedded Systems," IEEE Int. Conf. on Robotics and Automation, Mar. 2019.

[28] T. Senoh, N. Tetsutani, H. Yasuda and M. Teratani, "Revised Proposed Depth Estimation Reference Software (pDERS8.1)," ISO/IEC JTC1/SC29/WG11/MPEG2018/m45265.v3, Jan. 2019.

[29] G. Tech, H. Schwarz, K. Müller and T. Wiegand, "3D video coding using the synthesized view distortion change," Picture Coding Symposium, pp. 25-28, 2012.

[30] G. Tech, K. Müller, H. Schwarz and T. Wiegand, "Partial Depth Image Based Re-Rendering for Synthesized View Distortion Computation," IEEE Trans. on Circuits and Systems for Video Technology, vol. 28, no. 6, pp. 1273-1287, June 2018.

[31] K. Suehring, B. Li, V. Seregim, K. Sharman, G. Tech, A. Tourapis, "JCT-VC AHG report: Software development and software technical evaluation," ISO/IEC JTC1/SC29/WG11/MPEG2020/JCTVC-AL0003, Jan. 2020.

[32] Takeuchi, K., Okami, K., Ochi, D., et al.: Partial plane sweep volume for deep learning based view synthesis. ACM SIGGRAPH 2017.

[33] Li A., Fang L., Ye L., Zhong W., Zhang Q. (2020) Geometry-Guided View Synthesis with Local Nonuniform Plane-Sweep Volume. In: Zhai G., Zhou J., Yang H., An P., Yang X. (eds) Digital TV and Wireless Multimedia Communication. IFTC 2019. Communications in Computer and Information Science, vol 1181. Springer, Singapore.

[34] D. Marpe, H. Schwarz and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," IEEE Trans. on Circuits and Systems for Video Technology, vol. 13, no. 7, pp. 620-636, July 2003.

[35] J. Jung, B. Kroon, R. Doré, G. Lafruit and J. Boyce, "CTC on 3DoF+ and Windowed 6DoF (v2)," ISO/IEC JTC1/SC29/WG11/MPEG2018/N17726, July 2018.

[36] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," ITU-T Q.6/16, Doc. VCEG-M33, Mar. 2001.

[37] M. Orduna, C. Díaz, L. Muñoz, P. Pérez, I. Benito and N. García, "Video Multimethod Assessment Fusion (VMAF) on 360VR Contents," IEEE Trans. on Consumer Electronics, vol. 66, no. 1, pp. 22-31, Feb. 2020.

[38] Z. Wang, E. Simoncelli, and A. Bovik, "Multiscale structural similarity for image quality assessment," 37th Asilomar Conf. on Signals, Systems & Computers, vol. 2, pp. 1398–1402, 2003.

[39] S. Tian, L. Zhang, L. Morin and O. Déforges, "A Benchmark of DIBR Synthesized View Quality Assessment Metrics on a New Database for Immersive Media Applications," IEEE Trans. on Multimedia, vol. 21, no. 5, pp. 1235-1247, May 2019.

[40] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. "The unreasonable effectiveness of deep features as a perceptual metric," Proc. CVPR, 2018.

[41] B. Kroon, V. Kumar Malamal Vadakital, J. Jung, "Recommended pixel rate limits for the CTC for Immersive Video," ISO/IEC JTC1/SC29/WG11/MPEG2019/M49826, July 2019.

[42] E. Bosc, F. Racapé, V. Jantet, P. Riou, M. Pressigou, L. Morin, "A study of depth/texture bit-rate allocation in multi-view video plus depth compression," Annals of Telecommunications, Springer, 68 (11-12), pp.615-625, 2013.

[43] Y. Al-Obaidi, T. Grajek and M. Domański, "Quantization of Depth in Simulcast and Multiview Coding of Stereoscopic Video plus Depth Using HEVC, VVC and MV-HEVC," 2019 Picture Coding Symposium (PCS), Ningbo, China, pp. 1-5, 2019.

[44] K. Müller and A. Vetro, "Common Test Conditions of 3DV Core Experiments," Document JCT3V-G1100, JCTVC ITU-T SG16 WP3ISO/IECJTC1/SC29/WG11, San Jose, CA, USA, Jan. 2014.

Patrick Garus received the M. Sc and M. Ed. degrees from Rheinisch-Westfälische Technische Hochschule Aachen (RWTH Aachen), Aachen,Germany, in 2018 and 2019, respectively. He worked as a research assistant in the Institute of Communication Engineering (IENT), Aachen, Germany from 2015 to 2017 on synthesis of dynamic textures for video coding. He is currently pursuing the Ph.D. degree with Orange Labs and INRIA, Rennes, France. His current research interests include Immersive Video Coding, depth estimation, view synthesis and related machine learning-based technologies.

Félix Henry received the Dipl.-Ing. (M.Sc.) degree in telecommunications from Telecom ParisSud, Paris, France, in 1993, and the Ph.D. degree from Telecom ParisTech, Paris, in 1998 in the domain of wavelet image coding. He started his career in 1995 with the Canon Research Center, France, where he worked on still image compression and video coding. He participated actively in JPEG2000, HEVC and VVC standardization and is a co-inventor of more than 150 patents in the domain of signal processing. He is currently working as a research Engineer with Orange Labs and the b<>com National Research Institute, Rennes, France. His current research interests include Immersive Video Coding and Video Compression using Neural Networks.

Dr. Joel Jung received the habilitation degree in electrical engineering from Sorbonne University, Paris, France, in 2019, and the Ph.D. degree in electrical engineering from the University of Nice, Nice, France, in 2000. From 1996 to 2000, he was with the CNRS Laboratory, Sophia Antipolis, active in the improvement of video decoders based on the correction of compression and transmission artifacts. In 2000, he joined Philips Research France, Paris, as a Research Scientist in video coding, postprocessing, perceptual models, objective quality metrics, and low-power codecs. He worked at Orange Labs, France, from 2004 to 2020. He has contributed to the 2-D and 3-D video coding standard HEVC/3-D HEVC. He is currently a Principal Researcher with Tencent MediaLab, Palo Alto, USA. His current research interests include video quality evaluation for gaming content and immersive video content, contributing to ITU-T Study Group 12 and being chair of the immersive video focus group of MPEG Advisory Group 5 (AG5) on video quality assessment. In addition, he is involved in the standardization of immersive video codecs, as a co-chair of MPEG Immersive Video (MIV) group, dealing with coding, view synthesis, and depth estimation with six degrees of freedom.

Thomas Maugey graduated from Ecole Supérieure d'Electricité, Supélec, Gif-sur-Yvette, France in 2007. He received the M.Sc. degree in fundamental and applied mathematics from Supélec and Université Paul Verlaine, Metz, France, in 2007. He received his Ph.D. degree in Image and Signal Processing at TELECOM ParisTech, Paris, France in 2010. From October 2010 to October 2014, he was a postdoctoral researcher at the Signal Processing Laboratory (LTS4) of Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland. Since November 2014, he is a Research Scientist at Inria Rennes-Bretagne-Atlantique. He serves as an Assiociate Editor for EURASIP Journal on advances in signal processing. His research deals with image and video processing, 3D imaging and graph-based signal processing.

Christine Guillemot , IEEE fellow, is Director of Research at INRIA. She holds a Ph.D. degree from ENST (Ecole Nationale Superieure des Telecommunications) Paris, and an Habilitation for Research Direction from the University of Rennes. From 1985 to Oct. 1997, she has been with FRANCE TELECOM, where she has been involved in various projects in the area of image and video coding and processing for TV, HDTV and multimedia. From Jan. 1990 to mid 1991, she has worked at Bellcore, NJ, USA, as a visiting scientist. Her research interests are signal and image processing, and computer vision. She has served as Associate Editor for IEEE Trans. on Image Processing (from 2000 to 2003, and from 2014-2016), for IEEE Trans. on Circuits and Systems for Video Technology (from 2004 to 2006), and for IEEE Trans. on Signal Processing (2007-2009). She has served as senior member of the editorial board of the IEEE journal on selected topics in signal processing (2013-2015) and has been senior area editor of IEEE Trans. on Image Processing (2016-2020).