IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 12 January 2023, accepted 13 March 2023, date of publication 20 March 2023, date of current version 24 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3259899

RESEARCH ARTICLE

Supporting Wider Baseline Light Fields in JPEG Pleno With a Novel Slanted 4D-DCT Coding Mode

MURILO BRESCIANI DE CARVALHO[1], (Member, IEEE), CARLA L. PAGLIARI[2], (Senior Member, IEEE), GUSTAVO DE O. E. ALVES[3], (Member, IEEE), COLAS SCHRETTER[4,5], (Member, IEEE), PETER SCHELKENS[4,5], (Member, IEEE), FERNANDO PEREIRA[6,7], (Fellow, IEEE), AND EDUARDO A. B. DA SILVA[3], (Senior Member, IEEE)

[1]TET/CTC, Universidade Federal Fluminense, Niteroi 24220-900, Brazil
[2]PEE/PGED/IME, Instituto Militar de Engenharia, Rio de Janeiro 22290-270, Brazil
[3]PEE/COPPE/DEL/POLI/UFRJ, Universidade Federal do Rio de Janeiro, Rio de Janeiro 21941-901, Brazil
[4]Department of Electronics and Informatics, Vrije Universiteit Brussel, 1050 Brussels, Belgium
[5]Department of Digital and User-Centric Solution, imec, 3001 Leuven, Belgium
[6]Instituto Superior Técnico, Universidade de Lisboa, 1049-001 Lisboa, Portugal
[7]Instituto de Telecomunicações, 1049-001 Lisboa, Portugal

Corresponding author: Carla L. Pagliari (carla@ime.eb.br)

This work was supported in part by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) under Grant 310942/2021-7, in part by the Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro (FAPERJ) under Grant E-26/201.021/2022, and in part by the Research Foundation – Flanders (FWO) under Grant G0B3521N.

ABSTRACT Light fields are one of the emerging 3D representation formats with an effective potential to offer very realistic and immersive visual experiences. This capability comes at the cost of a very large amount of acquired data which practical use requires efficient coding solutions. This need was already addressed by the JPEG Pleno Light Field Coding standard for static light fields, which has specified two coding modes, named 4D-Transform and 4D-Prediction. While the first offers better compression performance for smaller baseline light fields, the second excels for larger baseline light fields. This paper intends to propose a novel light field coding mode, the Slanted 4D-Transform coding mode, which extends the 4D-Transform coding mode based on the conventional 4D-DCT, to offer better compression performance than both the available JPEG Pleno coding modes, independently of the baseline. The key idea is to apply first to each 4D block in the light field an adaptive, hierarchical geometric transformation, which makes the data in the block more energy-compaction friendly for the following 4D-DCT. The rate-distortion performance results show that the proposed Slanted 4D-Transform codec is able to outperform both the already standardized JPEG Pleno coding modes, showing BD-Rates gains of 31.03% and 28.30% for the 4D-Transform and 4D-Prediction modes, respectively, thus implying that a single coding mode can efficiently code all types of light fields.

INDEX TERMS JPEG Pleno standard, light field coding, 4D transform mode, 4D-DCT, slant transform.

I. INTRODUCTION

In recent years, a growing number and variety of multimedia services, applications, and devices have fundamentally changed the way humans communicate and interact with the surrounding world. This change has impacted all areas of human activity from personal communications, entertainment, and education to cultural heritage, health, business, and industry. The evolution of multimedia technology is largely driven by the goal of making multimedia experiences as realistic, faithful, rich, and immersive as real-life experiences. An easy check on this evolution may be performed using the so-called six degrees of freedom (6-DoF) associated with the three translational and three rotational movements that a human may exercise in real life. While 3-DoF virtual reality experiences are clearly one step forward over the usual two-dimensional (2D) experiences, there is still a lot of ground to cover.

6-DoF experiences have become the target for multimedia experiences where users should be able to also exercise the

The associate editor coordinating the review of this manuscript and approving it for publication was Binit Lukose.

three translational dimensions. For this to happen, there is the need for a more complete representation of the amount of light radiated at any position, in any direction, at any time instance, and at any wavelength, which corresponds to the well-known seven-dimensional (7D) plenoptic function [1]. As this function is able to represent all the light arriving at the human eye in the real world, it defines the target information that should ideally be acquired to replicate the real-world light and offer multimedia experiences with the same amount of visual realism and immersion. The most common models to approximate the plenoptic light information are light fields, point clouds, meshes, and holograms, and may involve different acquisition devices and their associated data processing challenges.

In this context, this paper will specifically address the light field representation model. Since this model is based on a large number of 2D scene perspectives/views, it allows for the light intensity, radiated from any position in the three-dimensional (3D) space, to be acquired/measured from multiple directions/perspectives as mandated by the plenoptic function. Since a wide variety of devices can be used for light field acquisition, from 2D regular camera arrays to lenslet cameras —where an array of micro-lenses overlays the sensor— light fields may have a large variation on the associated baseline, which is the distance between the camera centers from multiple scene perspectives/views. In practice, light field data carries both spatial and angular information about the light reaching the sensor, thus offering different viewpoints of the same 3D scene.

Naturally, the richness of the light field representation is associated with a huge amount of data whose practical usage, e.g., for storage and transmission, critically asks for efficient coding solutions. For the case of static light fields, an efficient codec should be able to explore not only the spatial and angular redundancies independently (as 2D data), but also the combined spatial-angular redundancy in 4D data space. To address these emerging compression efficiency needs, the Joint Photographic Experts Group (JPEG) [2] has completed in 2021 the first parts of the JPEG Pleno standard, notably Part 2, named Light Field Coding [3]. The JPEG Pleno standard intends to provide a framework to facilitate the capture, representation, and exchange of omnidirectional, depth-enhanced, point cloud, light field, and holographic imaging modalities [4]. It aims to define new tools for improved compression of the new plenoptic imaging modalities while providing advanced functionalities at the system level [4].

The JPEG Pleno Light Field Coding standard (Part 2) [3] specifies two coding modes for static light fields, known as 4D-Prediction mode (4D-PM) and 4D-Transform mode (4D-TM); while the latter is based on the 4D-DCT, the former exploits depth-based synthesis. The 4D-Transform mode specifies a 4D-native light field coding solution where the full light field redundancy, across the four dimensions, *i.e.*, across and within views, is jointly exploited. The 4D-Prediction mode relies on a depth maps-based warping process applied over some reference views coded with an available 2D codec, e.g., JPEG 2000, thus offering a rate-distortion (RD) performance that is very sensitive to the quality of the depth maps used for the view synthesis. As a plus, the 4D-Transform mode does not rely on any geometric data, e.g., depth maps, which may not be available or may be complex/expensive to obtain, notably with good enough quality.

While the two JPEG Pleno Light Field coding modes may code any light fields represented as a 2D array of 2D views [5], the 4D-TM offers better RD performance for more densely angular sampled light fields, such as lenslet light fields, but fails to perform well for light fields with wider baselines. On the other hand, the 4D-PM performs better for more sparsely angular sampled light fields at the expense of requiring good quality depth data, while not excelling for denser light fields. This performance difference was the key motivation for including two complementary coding modes in the JPEG Pleno Light Field Coding standard. Addressing this double weakness of the two coding modes is the goal of this paper.

This paper proposes an improved light field coding mode, named Slanted 4D-Transform mode (Slanted 4D-TM), which extends the current 4D-TM to perform better than both available JPEG Pleno Light Field coding modes, thus offering a single coding solution for the full range of light field baselines. The newly proposed Slanted 4D-TM targets to change the characteristics of the light field data fed to the 4D-DCT stage to reach better energy compaction. The attribute 'Slanted' —in practice meaning ''taking a diagonal course, direction, or path''— refers to the key intent of this new coding mode: the creation of a novel 4D-Transform, corresponding to the combination of the Slant transform (see Subsection IV-B) with the conventional 4D-DCT, able to adapt to the varying light field 4D block orientation characteristics. To understand the 4D-TM performance weakness for wider baselines, which the proposed coding mode addresses, it is helpful to analyze the light field Epipolar Plane Images (EPIs) [6], which are 2D spatial-angular slices of the light field. More precisely, if the light field is a matrix of views indexed by s – horizontal, and t – vertical, with each view being an image with coordinates u – horizontal, and v – vertical [5], then an EPI for coordinate (v_0, t_0) consists on the stacked horizontal scan lines of vertical coordinate v_0 for all views of horizontal coordinate t_0. For a light field of a Lambertian scene [7], an EPI is composed of diagonal lines of samples with constant intensity where each sample corresponds to the projection of a single 3D scene point. In this context, it is possible to derive a *depth-from-slope relationship* [6], which associates to each EPI slope a depth for the corresponding scene point. As a consequence, EPIs tend to consist of texture regions composed of straight lines with different inclinations, associated with the different depth values along the views. Fig. 1 shows the EPI for $(v_0, t_0) = (312, 0)$ for the *Bikes* light field and the top of Fig. 2 shows the EPI for $(v_0, t_0) = (256, 0)$ for the *Tarot* light field (see

FIGURE 1. Epipolar plane image (EPI) for $(v_0, t_0) = (312, 0)$ with dimensions 13×625 ($s \times u$) for the *Bikes* lenslet light field with dimensions $13 \times 13 \times 434 \times 625$ ($t \times s \times v \times u$).

FIGURE 2. Top: EPI for $(v_0, t_0) = (256, 0)$ with dimensions 17×1024 ($s \times u$) for the *Tarot* light field with dimensions $17 \times 17 \times 1024 \times 1024$ ($t \times s \times v \times u$). Bottom: EPI slanted so that EPI lines are almost vertical.

Fig. 13 and [2], [8]). While *Bikes* is a lenslet light field with an inherent small baseline for which the 4D-TM performs very well, *Tarot* is a larger baseline light field for which the 4D-TM performs poorly. Fig. 1 shows that the *Bikes* EPI is composed of almost vertical lines, and thus can be efficiently energy-compacted by a conventional separable and orthogonal 4D-DCT. On the other hand, the top of Fig. 2 shows that the EPI for the sparsely angular sampled *Tarot* light field presents larger slants/slopes, for which the conventional separable and orthogonal 4D-DCT cannot offer efficient energy-compaction.

In summary, the main novel idea proposed in this paper is to change the EPI slants in the 4D blocks to code by applying a geometric transformation before the conventional 4D-DCT stage, notably to make the slopes of the lines composing the EPIs as vertical as possible. In this way, a separable 4D-DCT, that is applied in the vertical and horizontal directions of the (s, u) and (t, v) EPIs, can have better energy compaction performance. An example of the effect of such geometric transformation referred to here as a Slant transform, can be seen at the bottom of Fig. 2, where the slanted EPI from the *Tarot* light field shows lines that are more vertical than the original EPI lines. This makes the slanted light field 4D blocks much more energy-compaction friendly for the conventional 4D-DCT. For the new EPI slants to be as 'vertical' as possible, the Slant transformation process is adaptive and hierarchical by dividing the 4D block size into smaller sizes whenever using a single slant slope is not efficient enough in terms of energy compaction, e.g., due to the presence of very different slants within the same block. Such a slant-then-transform process is equivalent to a separable slanted 4D-transform using two non-orthogonal directions, which is applied to blocks that are padded to create fully filled 4D hyper-parallelepiped blocks after the transformation. In this way, the coding process adapts and compensates for the slant slopes typically associated with larger baselines, thus making the full coding process efficient for all types of light field baselines.

While it is common to characterize light fields by their baselines, i.e., the distance between camera optical centers, this is a simplistic approach. The depth range that may be observed changes with the scene geometry since objects can be at different depth planes. Objects closer to the camera (shallow depth) in wide baseline parallel camera setups have high disparity values, while those farther away from the camera have low disparity values. Thus, it is not possible to simplistically generalize that wide baseline systems will always generate light fields with large disparities as it depends on the 3D scenery being imaged. This means that scene geometry may also be a challenge to any light field coding solution targeting to exploit the full 4D redundancy. Therefore, the overall space-view structure has to be taken into account when designing a light field coding method. As a consequence, this work will consider light fields with a large range of space-view redundancy, notably for RD performance assessment, to address the full challenges of different light field baselines.

The new light field coding mode proposed in this paper outperforms the already available 4D-Transform and 4D-Prediction light field coding modes for all baselines, thus fulfilling the target of offering a single, efficient coding mode, which is rather 'agnostic' to the specific baseline. This work uses the light fields recently selected for the JPEG Pleno Light Field Quality Assessment Common Test Conditions [9] developed in the context of the JPEG Pleno Light Field Quality Assessment Call for Contributions [10], achieving rate reductions measured as BD-Rates that amount to 31.03% and 28.30% relative to the ones of the 4D-TM and 4D-PM, respectively.

To achieve its goals, this paper is organized as follows: Section II briefly reviews the most relevant static light field coding solutions in the literature and the JPEG Pleno Light Field Coding standard, while Section III presents in detail the 4D-TM due to its specific relevance for this paper. Section IV proposes the Slanted 4D-TM which has the potential to become an additional JPEG Pleno Light Field coding mode with the advantage of offering better performance than both current standard coding modes, without breaking backward compatibility. Finally, Section V presents the performance assessment using appropriate materials, conditions, metrics, and benchmarks, and Section VI concludes the paper.

II. STATIC LIGHT FIELD CODING REVIEW

A large number of static light field coding methods have been proposed in recent years, spanning from adaptations of conventional 2D image/video codecs to those specially designed for light field data. A comprehensive survey on light field coding methods is presented in [11], especially addressing densely angular sampled light fields. For precision, angular sampling density refers to the ratio between the number of sampled viewpoints and the corresponding field of view of the light field [12], [13]. A light field coding classification taxonomy is proposed in [14] using two main dimensions, notably micro-images (MI) versus perspective-images (PI) based coding, where an MI consists of all samples (light rays) corresponding to a given 3D-space point, while a PI corresponds to all samples from a specific light field point of view. Another taxonomy dimension differentiates between texture-only versus texture+geometry light field

coding. While texture-only light field coding relies only on the light field texture samples, texture+geometry light field coding relies both on the light field samples as well as some geometry-related information, such as depth maps.

A. MOST RELEVANT CODING METHODS

The most relevant static light field coding methods in the literature are here reviewed guided by the taxonomy proposed in [14], starting with the texture-only and followed by the texture+geometry light field coding methods. While the data fed to most light field coding methods are PIs (a.k.a. sub-aperture images or views), some light field codecs code MIs. Some texture-only light field coding methods are similar to standard image codecs since they directly code the raw light field data arranged as a (large) MI or PI image. Spatial correlation is usually exploited using suitable well-known intra-coding tools. In [15], the High-Efficiency Video Coding standard - Screen Content Coding (HEVC-SCC) extension [16], [17] is adopted to code the light field data organized as a single, large 2D matrix of MIs. The HEVC-SCC *intra block copy mode* proved useful for light field coding as MIs tend to include repetitive patterns.

In other texture-only coding methods, the light field MIs or PIs can be sequenced as a pseudo-video with a specific scan order aiming at maximizing the correlation between the frames of the resulting pseudo-video. Well-known video coding standards, such as H.264/AVC [18], HEVC [19] and Versatile Video Coding (VVC) [20], have been used to code light field data arranged as a pseudo-video. Different scanning patterns have been proposed to improve the RD performance by better exploiting the redundancy between MIs or PIs; good examples can be found in [21] and [22]. In [23], an HEVC-based codec mixes hybrid prediction structures for MIs and PIs, depending on which is more efficient for different parts of the light field. In [24], light field image views are grouped into the so-called Macro View Images (MVIs), which are groups of independent 3×3 views. The MVIs are encoded by HEVC [19] or VVC [20] codecs with the central view used as a reference to compress its adjacent neighboring views using a hierarchical reference structure. This coding structure provides random access. The best RD performance is obtained using VVC, leading to the MVI-VVC codec.

Along a similar line, it is possible to create a multi-view pseudo-video from the light field PIs and encode it using a multiview video coding standard, such as the Multi-View Video Coding (MVC) standard [25] or the MV-HEVC standard, the HEVC multi-view extension [19], [26]. Good examples of such methods are proposed in [27], [28], and [29]. These coding methods differ essentially in the way the light field PIs are organized to assemble the multi-view pseudo-video. The work in [30] adds an additional module to the MV-HEVC-based pipeline with the light field data partitioned into key views and a set of *decimated views*. While the former are coded using MV-HEVC, the latter are predicted using a shearlet transform-based extrapolation, and the generated residual is also coded with MV-HEVC.

Other texture-only light field coding methods do not rely on pseudo-videos. For example, [31] proposes a light field coding method that associates graphs and dictionary learning to remove the structural redundancy between PIs. In this case, just a few key views are encoded using HEVC [19], while the remaining non-key views are reconstructed via the graph adjacency matrix learned from the angular patch. In [32], selected reference views are coded with HEVC while the remaining views are predicted using a linear combination of the decoded reference views and their residuals encoded using Principal Component Analysis (PCA). Along a different line, the light field coding method in [33] adopts a graph-based lifting transform, thus introducing an intra-prediction approach that operates on the non-demosaicked lenslet PIs.

Most texture+geometry coding methods rely on depth information to enhance the compression performance. Commonly, such methods use depth-assisted view estimation to recover the entire light field from a small subset of previously coded views. One notable example is given by the JPEG Pleno light field 4D-PM codec described in detail in Section II-B. A different approach to estimate and exploit the depth information is given by the so-called Fourier Disparity Layer (FDL) representation [34], where a representation of the light field based on disparity layers is computed using least square regression in the 4D Fourier domain; this offers a flexible and efficient light field rendering. Texture+geometry light field coding methods have been proposed based on the FDL representation. In [35], the light field is partitioned into subsets of PIs. The first subset is coded as a video sequence using HEVC while the remaining subsets are iteratively estimated using an FDL representation, with the resulting prediction residues coded with HEVC. A different approach is proposed in [36], where the light field FDL representation is coded instead of its PIs using a binary hierarchical HEVC-based encoder (FDL-hierarchical).

Following its success in image coding, learning-based approaches have recently been adopted for light field coding. A notable example of a texture+geometry coding method using a learning-based approach can be found in [37]. A depth-adaptive convolutional neural network (CNN) is proposed, where two new intra-prediction modes are combined with the 35 HEVC standard intra-prediction modes. For each HEVC's Coding Unit block size, the two new intra-prediction modes select different reference blocks according to the depth of the current block. In [38], a 3D-CNN autoencoder is proposed, assisted by a disparity estimation process. This end-to-end texture+geometry trainable model does not require any hand-crafted features for learning the disparity. The proposed model learns a better bit allocation scheme by jointly optimizing for rate and distortion. Learning-based coding is currently a very hot research line for multimedia coding and standardization bodies are starting to

pay attention, e.g., JPEG AI [39], [40] and JPEG Pleno Point Cloud Coding [41].

B. JPEG PLENO LIGHT FIELD CODING STANDARD

The JPEG Pleno standard includes several parts dedicated to plenoptic imaging representation, notably its Part 2, named JPEG Pleno Light Field Coding, which specifies the codestream syntax, decoding process, and associated metadata for light field coding [3]. In Part 2, two JPEG Pleno Light Field Coding standard modes are specified, notably the 4D-Transform coding Mode (4D-TM) and 4D-Prediction coding mode (4D-PM).

The JPEG Pleno Light Field 4D-TM includes five main modules, notably block partitioning, a transform, a quantizer, a symbol generator, and an entropy encoder. It adopts a multiscale 4D-DCT and a hexadeca-tree-oriented bit-plane clustering approach, relying on an adaptive 4D segmentation scheme to partition the light field data into a set of disjoint four-dimensional (4D) sub-blocks of varied sizes. The set of 4D sub-blocks forms a 4D Partition of the original light field, which is represented by a segmentation tree where each leaf node is associated with a single 4D sub-block. Each 4D sub-block is transformed by a separable 4D-DCT transform and the bit-planes of its coefficients are subsequently encoded using an hexadeca-tree structure. Due to its importance for this paper, the 4D-TM will be described in more detail in Section III, and a thorough description can be found in [42].

The JPEG Pleno Light Field 4D-PM is based on the Warping and Sparse Prediction (WaSP) coding method detailed in [43]. This coding mode selects some views, PIs, as reference views, in which texture and depth are coded using the (royalty-free) JPEG 2000 standard [44] or other JPEG standardized codecs. The remaining/intermediate views are coded by exploiting the sample correspondences between them and the reference views, which are obtained from the associated depth maps and camera parameters. At this stage, the samples from the reference are warped into the intermediate view positions, followed by a prediction phase where the multiple warped views are merged into a full intermediate view using least-squares optimal predictors over a set of occlusion-based regions [45]. A variation of the 4D-PM is presented in [45], known as WaSPR (Warping and Sparse Prediction on Regions) using the HEVC [19] standard to code the reference views. Since the JPEG Pleno Light Field 4D-PM relies on depth data that feeds the warping and inpainting modules, the depth information needs to be estimated (or acquired) and encoded. In addition, camera parameters should be signaled.

It is important to note that the 4D-PM RD performance is quite sensitive to the accuracy of the depth data, which plays a key role in the quality of the predictions. Therefore, while the 4D-TM does not need any geometry/depth information, the 4D-PM performance may suffer from poor depth maps, e.g., inconsistencies. In addition, the 4D-PM employs legacy codecs (e.g., JPEG 2000 or other standardized JPEG codecs) while the 4D-TM does not employ any legacy image codec. Though the 4D-TM outperforms the 4D-PM in RD terms for narrow baseline light fields, the opposite happens for light fields with wider baselines. This different compression performance behavior was the key reason for including two (complementary) coding modes in the JPEG Pleno Light Field Coding standard.

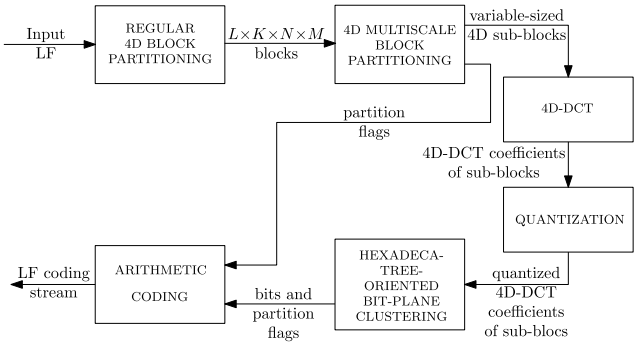


FIGURE 3. JPEG pleno light field coding 4D-TM architecture according to [42].

The JPEG Pleno Part 2 amendment [46] defines two profiles and several levels for JPEG Pleno Light Field Coding. While the Baseline Block-based profile comprises all coding tools included in the 4D-TM, the Baseline View-based profile comprises all coding tools included in the 4D-PM. This way, a light field decoding device may be compliant with any of the two profiles or with the two profiles simultaneously. For each profile, four levels have been defined. For the Baseline Block-based profile the levels depend on the *Maximum number of samples* and *Maximum block dimension*. For the Baseline View-based profile the levels depend only on the *Maximum number of samples*. Here a sample corresponds to a single component value per channel, texture, or depth [46].

III. JPEG PLENO LIGHT FIELD CODING 4D-TRANSFORM MODE

The JPEG Pleno Light Field Coding 4D-TM [3], [46] is a coding method that is part of the MuLE (**Mu**ltidimensional **L**ight Field **E**ncoder) codec family, described in detail in [42]. More specifically, the JPEG Pleno Light Field 4D-TM referred to as MuLE-MTH (**M**ultiscale **T**ransforms and **H**exadeca-tree-oriented bit-plane clustering) in [42] is a multiscale and hexadeca-tree-oriented bit-plane clustering codec, already briefly described in Section II-B. Fig. 3 illustrates the JPEG Pleno Light Field 4D-TM coding architecture according to [42]. In the sequel, the essentials of the JPEG Pleno Light Field 4D-TM will be presented based on this figure. For a more detailed description, the reader is referred to [42].

The light field data input to the 4D-TM codec is interpreted as a two-dimensional (2D) array of 2D images (associated with spatial information) called *views* (associated with angular information) [5]. The views are sampled with

sampling intervals Δt and Δs and are indexed by view integer coordinates (l, k), that is, $l\Delta t$ represents a vertical displacement of the camera for that particular view and $k\Delta s$ represents a horizontal displacement. The samples inside each view are indexed by the integer spatial coordinates (n, m), with n indexing the vertical direction and m the horizontal direction. Therefore, the tuple (l, k, n, m) indexes a single unique sample in the light field. The JPEG Pleno Light Field 4D-TM architecture in Fig. 3 works as follows:

- *Regular 4D Block Partitioning* - Initially, the light field is segmented into 4D blocks of fixed size $(L \times K \times N \times M)$, where the L, K, N and M values can be the same as the full light field dimensions if desired. As observed in [42], the RD performance tends to increase as the initial block size increases, but so does the computational complexity; therefore, the choice of the block size involves a trade-off between compression efficiency, complexity, and also random access. After this regular partitioning, the initial 4D blocks are independently and sequentially encoded.
- *4D Multiscale Block Partitioning* - Each $L \times K \times N \times M$ light field block is recursively partitioned into variable-sized 4D sub-blocks so that an RD cost, $J = D + \lambda R$ is minimized, where D is the distortion of the $L \times K \times N \times M$ block when reconstructed by the decoder and R is the associated coding rate. Partition flags signal the tree that defines the dynamic block partitioning.
- *4D-DCT* - A separable conventional 4D-DCT is applied to each variable-sized sub-block to generate the corresponding 4D-DCT transform coefficients.
- *Quantization* - The target quality and rate are controlled by the quantization of the 4D-DCT coefficients. In JPEG Pleno Light Field 4D-TM, the amount of quantization is set by the number of bit-planes used to represent the transform coefficients. Note that, as defined in [42], the lowest bit-plane contains the least significant coefficient bits, and the highest bit-plane contains the most significant ones.
- *Hexadeca-tree-oriented Bit-plane Clustering* - This module efficiently encodes clusters of leading zeros of the 4D-DCT coefficients, also optimizing the RD cost [42]. The process starts with the binary representation of the quantized 4D-DCT coefficients using a bit-depth corresponding to the adopted quantizer, which is determined using an RD criterion. For each hexadeca-tree node, a 4D block is recursively partitioned into 16 4D sub-blocks by splitting each one of its four dimensions into two approximately equal-sized segments. Partition flags are encoded to indicate the partitioning of the hexadeca-tree. This allows for performing tree clustering of the non-significant (quantized as zero) 4D transform coefficients and positioning of the significant ones [42].
- *Arithmetic Coding* - The flags defining the partition tree, the flags defining the hexadeca-tree and the 4D-DCT coefficients bits are interleaved and then encoded using a context-adaptive binary arithmetic encoder.

Decoding the light field is a straightforward process since the decoder receives the flags as well as coefficient bit-planes and simply recovers the variable-sized sub-blocks of 4D-DCT coefficients. The inverse 4D-DCT is then applied to each block of coefficients to generate the reconstructed light field.

From the above, it may be concluded that the JPEG Pleno Light Field Coding 4D-TM has the following strengths:

- Very good RD performance for narrow baseline light fields, such as lenslets;
- 4D-native, that is, able to jointly exploit the correlations among the four light field dimensions;
- Does not rely on any geometric data, notably depth maps, that would need to be acquired or estimated at the encoder side and coded to be made available to the decoder, which makes the 4D-TM usage straightforward;
- Provides random access to the coded 4D blocks since the light fields are partitioned into equal-sized 4D blocks that are independently coded (see Fig. 3);
- Low computational decoder complexity, since it just has to recover the 4D block partition and the hexadeca-tree from the encoded flags and bits, and compute the inverse 4D-DCT;
- Apparently, royalty-free.

On the other hand, the main JPEG Pleno Light Field 4D-TM weaknesses are:

- RD performance tends to degrade significantly as the light field baseline increases since the redundancies along the interview directions decrease, thus decreasing the 4D-DCT compaction efficiency [42].
- Good RD performance is usually only for lenslet light fields; for larger baseline light fields, the 4D-PM offers better RD performance.

The next section will propose a novel Slanted 4D-Transform coding mode, which is still a 4D-native mode not dependent on depth maps or any geometry data while offering competitive RD performance, irrespective of the light field baseline. Thus, this new coding mode keeps the main JPEG Pleno Light Field 4D-TM strengths while addressing and overcoming its main weaknesses.

IV. SLANTED 4D-TRANSFORM CODING MODE

As seen before, the main weakness of the JPEG Pleno Light Field 4D-TM is related to its poor performance for light fields with wide baselines. By comparing its RD performance [42], [47] with other coding methods such as the JPEG Pleno Light Field 4D-PM [47], it is clear that the 4D-TM RD performance is quite good for light fields with narrow baselines but it degrades fast as the angular sampling increases. As pointed out in the discussion associated with Figs. 1 and 2 in Section I, some insight about this behavior can be obtained by observing the epipolar plane images (EPIs) of the 4D blocks to code.

Since the 4D-TM is DCT based, it is clear from the EPIs that a direct application of a separable DCT over the 4D blocks represented in Fig. 1 and the top of Fig. 2 would produce very different results. When applying the 4D-DCT to the 4D blocks corresponding to the EPI in Fig. 1, higher energy compaction is expectable; this is due to the lack of high-frequency components along the view axis (s) when compared to the high frequencies present along the s axis in Fig. 2; this is a direct consequence of the lines with non-vertical slopes present in this EPI.

A possible approach to enhance the 4D-DCT energy compaction performance in this context, and as a consequence of the 4D-TM compression performance, is to apply a geometric transformation to the 4D block prior to the application of the 4D-DCT, in an attempt to change the slopes of the EPI lines such as those at the top of Fig. 2 so that they become mostly vertical, as those in Fig. 1 and at the bottom of Fig. 2. This way a separable 4D-DCT applied to the vertical and horizontal interview directions would provide better energy compaction. A geometric transformation with this type of behavior is here called a *Slant* transform. This type of transformation has been applied in [48] in the context of a light field depth estimation solution.

A. ARCHITECTURE AND WALKTHROUGH

Fig. 4 shows the overall architecture of the proposed Slanted 4D-TM codec. The changes to the previously presented JPEG Pleno Light Field 4D-TM standard codec are highlighted by the gray blocks while the remaining blocks are the same as in Fig. 3 since the newly proposed coding mode is an extension of the already available 4D-TM (which would still exist for backward compatibility purposes). The proposed Slanted 4D-TM coding mode proceeds as follows, outlining the additions to the original 4D-TM codec signaled in Fig. 3:

- *Slant Transform* - This module has the objective of geometrically transforming a 4D block of a light field by performing the proposed Slant operation. Fig. 5(b) shows an example of the input $s \times u$ EPI at line 644 (highlighted in red in Fig. 5(a)) and Fig. 5(c) shows the resulting EPI after the Slant transform is applied. More details on the Slant Transform module are provided in Subsection IV-B.
- *Trimming* - During the Slant tree optimization process, based on successive slanting and partitioning, it is possible that a 4D block is slanted in one direction in a specific recursion level, and, at the next recursion level, some of the 4D sub-blocks resulting from block partitioning are further slanted in the opposite direction. This might lead to a hyper-rectangular region where all samples are empty. In this case, these samples should be discarded (trimmed). More details on the trimming operation are given in Subsection IV-C.
- *Padding* - The padding module is needed because the 4D-DCT must be applied to a hyper-rectangular $t \times s \times v \times u$ 4D block and not to any other shape.

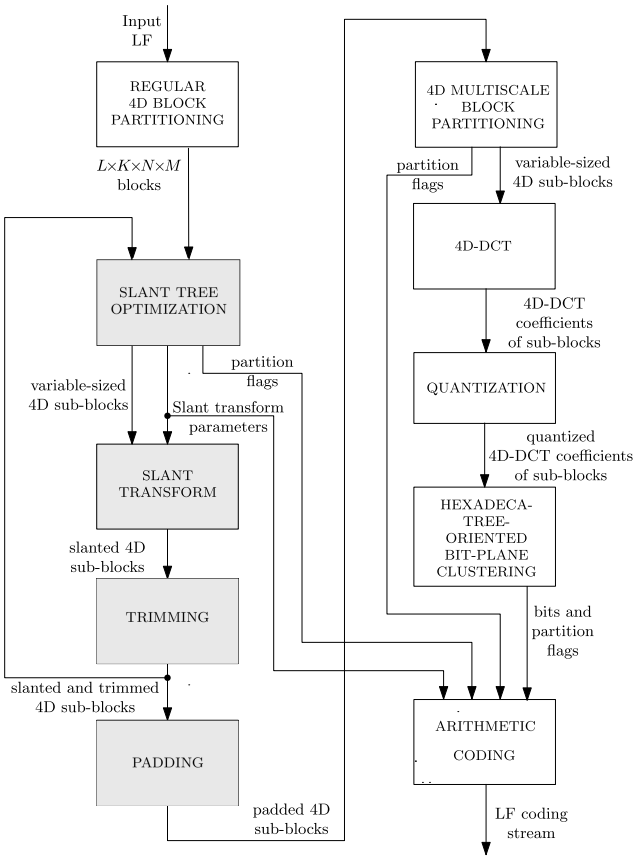


FIGURE 4. Slanted 4D-TM coding architecture.

FIGURE 5. (a) *Laboratory1* light field top left view, (b) $s \times u$ EPI at line 644 (highlighted in red), (c) slanted EPI, and (d) slanted and padded EPI - with stretched s dimension for better visualization.

However, if a Slant transform is applied to a 4D block, it is sheared into a 4D hyper-parallelepiped that cannot be directly processed by the 4D-DCT. In this context, the sheared 4D hyper-parallelepiped must be padded along the t, s, v and u directions, so that it becomes a $t \times s \times v \times u$ 4D hyper-rectangle. More details on the padding operation are given in Subsection IV-D.

- *Slant Tree Optimization* - To maximize the compression efficiency of the 4D-DCT applied after padding, each hyper-rectangular region of the light field may have a different Slant transform. Therefore, the $L \times K \times N \times M$ 4D blocks output by the regular 4D block partitioning module may be further partitioned so that after each individual 4D block has been subjected to its optimal Slant transform, trimmed, and padded, the set of slanted 4D blocks reaches optimized 4D-DCT-based compression efficiency. Thus the Slant tree optimization module has the objective to find, recursively, the optimal 4D block partitioning and associated optimal Slant transform parameterization for each resulting 4D sub-block. More details on the Slant Tree Optimization module are given in Subsection IV-E.

Finally, the slanted, trimmed and padded 4D blocks are encoded by the standard JPEG Pleno Light Field 4D-TM, now with characteristics that maximize the final RD performance. The modules outlined in this subsection will be further detailed in the following subsections.

B. SLANT TRANSFORM

As mentioned before, the main goal of the Slant Transform module is to define the optimal geometric transformation for each 4D sub-block such that the Slanted 4D block at its output maximizes the 4D-DCT compression efficiency. This subsection proposes a new design for the Slant transform aiming to give adaptability capabilities to the 4D-TM codec in terms of the light field 4D block geometric characteristics. This way, the 4D-TM codec may be used to code light fields with a large range of baselines since the light field 4D blocks are previously processed to become geometrically more suitable for the 4D-DCT, with the target benefits in terms of increased compression efficiency. While the Slant transform itself is available in the literature, its detailed formulation and application to light field coding with an adaptability target — notably by applying the 4D Slant transform as a separable combination of two 2D Slant transforms to the EPIs of the discrete light field— is novel and at the heart of the proposed light field coding method.

The Slant transform can be better understood if first explained with the continuous light field $\mathcal{L}(t, s, v, u)$ along the dimensions t, s, v and u, where t and s are the vertical and horizontal positions of a view, respectively, and v and u are the vertical and horizontal coordinates of a sample within a view. Before specifying the proposed Slant transform process, it is important to properly define the epipolar plane image, a slice of the light field parallel either to the $t \times v$ or to the $s \times u$ planes, repeated here for convenience:

- A $t \times v$ EPI associated to the coordinates (s_0, u_0) is the set of points $\mathcal{L}(t, s_0, v, u_0)$.
- A $s \times u$ EPI associated to the coordinates (t_0, v_0) is the set of points $\mathcal{L}(t_0, s, v_0, u)$.

Moreover:

1) The image of a 3D-space point P in a lenslet or parallel HDCA light field $\mathcal{L}(t, s, v, u)$, such as those used in [8], is a 2D plane in the $t \times s \times v \times u$ 4D-space. This 2D plane intercepts the $s \times u$ and $t \times v$ EPI images of the light field in straight lines [5], [6], [12].
2) The angular coefficients of these straight lines are equal and a function of the depth of the 3D-space point P [7], that is:

$$\frac{\delta u}{\delta s} = \frac{\delta v}{\delta t} = \sigma. \tag{1}$$

where σ is a function of the depth of P.

The continuous Slant transform with slope parameter σ and base view (t_b, s_b) is the transform that, when applied to a continuous light field, generates a transformed light field $\mathcal{L}^{\sigma;t_b,s_b}(t, s, v, u)$. This transform is such that the base view is not changed, and there is a depth z such that the image of a 3D-space point P with depth z in $\mathcal{L}^{\sigma;t_b,s_b}(t, s, v, u)$ has the same (v, u) coordinates in all view positions of the light field. In addition, these coordinates are equal to those of the image of the 3D-space point P in the base view (t_b, s_b) in the original light field $\mathcal{L}(t, s, v, u)$. This implies that in $\mathcal{L}^{\sigma;t_b,s_b}(t, s, v, u)$:

- There is a depth z such that the images of 3D-space points at depth z will have zero disparity.
- The intersection of the image of a 3D-space point P at the above depth z with the $t \times v$ and $s \times u$ EPIs are straight lines that are orthogonal to the t and s axes, respectively, that is,

$$\frac{\delta u}{\delta s} = \frac{\delta v}{\delta t} = 0. \tag{2}$$

- $\mathcal{L}^{\sigma;t_b,s_b}(t_b, s_b, v, u) = \mathcal{L}(t_b, s_b, v, u), \forall\, (v, u)$.

From the above, the continuous Slant transform with slope parameter σ and base view (t_b, s_b) can be expressed as

$$\begin{aligned}\mathcal{L}^{\sigma;t_b,s_b}(t, s, v, u) &= \mathcal{S}^{\sigma;t_b,s_b}\{\mathcal{L}\}(t, s, v, u) \\ &= \mathcal{L}(t, s, v + (t - t_b)\sigma, u + (s - s_b)\sigma),\end{aligned} \tag{3}$$

where σ is defined in Eq. (1).

By examining Eq. (3), it is possible to derive an important property of the Slant transform – it is separable. This can be readily verified by defining the 2D Slant transforms of a $t \times v$ EPI as $\mathcal{S}^{\sigma}_{t\times v}\{\cdot\}$ and of a $s \times u$ EPI as $\mathcal{S}^{\sigma}_{s\times u}\{\cdot\}$ as

$$\mathcal{S}^{\sigma;t_b}_{t\times v}\{\mathcal{L}\}(t, s, v, u) = \mathcal{L}(t, s, v + (t - t_b)\sigma u), \tag{4}$$

$$\mathcal{S}^{\sigma;s_b}_{s\times u}\{\mathcal{L}\}(t, s, v, u) = \mathcal{L}(t, s, v, u + (s - s_b)\sigma). \tag{5}$$

Then, from Eqs. (3), (4) and (5) it results that

$$\mathcal{S}^{\sigma;t_b,s_b}\{\mathcal{L}\}(t, s, v, u) = \mathcal{S}^{\sigma;s_b}_{s\times u}\left\{\mathcal{S}^{\sigma;t_b}_{t\times v}\{\mathcal{L}\}\right\}(t, s, v, u), \tag{6}$$

that is, the 4D continuous Slant transform with slope parameter σ can be computed by first applying a continuous 2D Slant transform with slope parameter σ to all the $t \times v$ EPIs of the light field and then applying a continuous 2D Slant transform with slope parameter σ to all the $s \times u$ EPIs of the first Slant transform result. Due to this key property, several parts of this work deal with just the 2D Slant transforms that are applied to the EPIs, since from Eq. (6) they are enough to implement a 4D Slant transform.

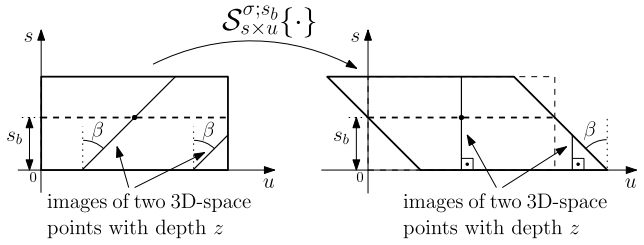


FIGURE 6. Illustration of the continuous 2D $s \times u$ slant transform $\mathcal{S}^{\sigma;s_b}_{s\times u}\{\cdot\}$ with slope parameter σ and base view s_b. Note that $\tan\beta = \delta u/\delta s = \sigma$ (Eq. (1)).

A continuous 2D $s \times u$ Slant transform $\mathcal{S}^{\sigma;t_b}_{t\times v}\{\cdot\}$ with slope parameter σ and base view s_b is represented in Fig. 6, where the original and slanted light fields are shown on the left and right, respectively. The images of 3D-space points with depth z on the $s \times u$ EPI, straight lines with inclinations $\delta u/\delta s = \sigma = \tan\beta$ (see Eq. (1)), are shown in the original light field. These straight lines are mapped by the 2D Slant transform $\mathcal{S}^{\sigma;t_b}_{t\times v}\{\cdot\}$ to straight lines on the $s \times u$ EPI of the slanted light field that is orthogonal to the u axis. This implies that a rectangular region on the original $s \times u$ EPI is mapped by the 2D Slant transform into a parallelogram whose side is also inclined by β, but in the opposite direction of the straight line in the original EPI. Note that a view at coordinate s_b, indicated by the dashed horizontal line in the $s \times u$ EPI, is not changed by the Slant transform.

As for most relevant cases, this work is interested in discrete light fields, which are functions of the sampling indices $l, k, m, n \in \mathbb{Z}$. These light fields are obtained by sampling the continuous variables t, s, v and u with sampling intervals Δt, Δs, Δv and Δu. The discrete light field $LF(l, k, n, m)$ can be expressed as a function of the continuous light field $\mathcal{L}(t, s, v, u)$ as

$$LF(l, k, n, m) = \mathcal{L}(t_0 + l\Delta t, s_0 + k\Delta s, v_0 + n\Delta v, u_0 + m\Delta u), \quad (7)$$

that is, given a light field sample at position (t, s, v, u), the integer sampling indices (l, k, n, m) are computed as

$$l = \left\lfloor \frac{t - t_0}{\Delta t} \right\rceil, \quad k = \left\lfloor \frac{s - s_0}{\Delta s} \right\rceil, \quad n = \left\lfloor \frac{v - v_0}{\Delta v} \right\rceil, \quad m = \left\lfloor \frac{u - u_0}{\Delta u} \right\rceil, \quad (8)$$

where $\lfloor\cdot\rceil$ is the rounding operation.

Then, applying the sampling implied by Eq. (7), Eq. (3) becomes

$$\mathcal{L}^{\sigma;t_b,s_b}(t_0 + l\Delta t, s_0 + k\Delta s, v_0 + n\Delta v, u_0 + m\Delta u) = \mathcal{L}(t', s', v', u'), \quad (9)$$

where

$$\begin{aligned} t' &= t_0 + l\Delta t, \\ s' &= s_0 + k\Delta s, \\ v' &= v_0 + n\Delta v + (l - l_b)\Delta t\sigma, \\ u' &= u_0 + m\Delta u + (k - k_b)\Delta s\sigma, \end{aligned} \quad (10)$$

and l_b and k_b are the indexes corresponding to the base view.

If Eq. (8) is applied to compute the set of indexes (l', k', n', m') corresponding to the sample at (t', s', v', u') of the original continuous light field as in (10), it comes

$$\begin{aligned} l' &= l, \\ k' &= k, \\ n' &= n + \left\lfloor (l - l_b)\frac{\Delta t}{\Delta v}\sigma \right\rceil, \\ m' &= m + \left\lfloor (k - k_b)\frac{\Delta s}{\Delta u}\sigma \right\rceil. \end{aligned} \quad (11)$$

If a 4D Slant transform with slope parameter σ and base view indexes (l_b, k_b) is to be applied to a discrete 4D block B with dimensions $L \times K \times N \times M$, then the 4D Slant transform has to be applied to each point (l', k', n', m') of B, to shift these points to the coordinates (l, k, n, m) of the slanted 4D block $B^{\sigma;l_b,k_b}$, as specified by Eq. (11). The algorithm to perform the Slant transform of a discrete 4D block B is presented as Algorithm 1.

Algorithm 1 Slant Transform of a Discrete 4D Block

1. *Inputs*:
   - *Slant transform parameters:* slope parameter σ and base view indexes (l_b, k_b).
   - *Input:* 4D block B, contained in the region $l \in \mathbb{N} \cap [0, L-1]$, $k \in \mathbb{N} \cap [0, K-1]$, $n \in \mathbb{N} \cap [n_L, n_R]$ and $m \in \mathbb{N} \cap [m_L, m_R]$.
2. *Output:*
   - *Slanted 4D block:* $B^{\sigma;l_b,k_b}$.
3. *Initialization:*
   - $B^{\sigma;l_b,k_b}(l, k, n, m) =$ empty, $\forall\, l, k, n, m$.
4. *Generation of the slanted 4D block:*

```
for l' ∈ ℕ ∩ [0, L − 1]
  for k' ∈ ℕ ∩ [0, K − 1]
    for n' ∈ ℕ ∩ [n_L, n_R]
      for m' ∈ ℕ ∩ [m_L, m_R]
        Compute (l, k, n, m) using Eq. (11);
        B^{σ;l_b,k_b}(l, k, n, m) = B(l', k', n', m')
      end
    end
  end
end
```

From Step 4. of Algorithm 1 and Eq. (11), it is possible to observe that the coordinates of the non-empty samples of view (l, k) of the slanted 4D block $B^{\sigma;l_b,k_b}$ are integers such

that

$$n_L - \left\lfloor (l - l_b)\frac{\Delta t}{\Delta v}\sigma \right\rceil \leq n \leq n_R - \left\lfloor (l - l_b)\frac{\Delta t}{\Delta v}\sigma \right\rceil,$$
$$m_L - \left\lfloor (k - k_b)\frac{\Delta s}{\Delta u}\sigma \right\rceil \leq m \leq m_R - \left\lfloor (k - k_b)\frac{\Delta s}{\Delta u}\sigma \right\rceil. \quad (12)$$

The inequalities in (12) make clear that the shape of the result $B^{\sigma;l_b,k_b}$ of a Slant transform applied to a 4D block B is a sheared 4D hyper-parallelepiped. Its intersection with a plane parallel to the $k \times m$ axes looks like the parallelogram illustrated on the right side of Fig. 6. Note that, in this figure, the left side shows the intersections with the images of two 3D-space points P_1 and P_2 having the same depth z, one that crosses all views, and another that does not cross all views. The former corresponds to a 3D-space point P_1 that is visible in all views, while the latter corresponds to a 3D-space point P_2 that is not visible in all views. On the slanted 4D block, the samples corresponding to the views in which a 3D-space point P_2 is not visible in the original 4D block are those empty.

Also, from Step 4. of Algorithm 1 and Eq. (11), it is possible to observe that the samples of the slanted 4D block are just samples of the original 4D block shifted to other integer coordinates. Therefore, no interpolation must be performed as the sample values are not altered by the Slant transform. In addition, no two samples having different coordinates in the original 4D block are mapped to the same coordinates in the slanted 4D block. This implies that the Slant transform, as defined in Algorithm 1, is exactly *reversible*. This is a convenient property in coding since reversibility allows lossless compression.

Finally, it is important to point out that the slope parameter σ of the Slant transform is depth-related in the sense that there is a depth z related to the slope parameter σ such that the images of 3D-space points at this depth will possess the same (n, m) coordinates for all views (l, k). However, there is absolutely no need for any depth map to encode a light field using the proposed coding method. This is so because a Slant transform for a whole 4D block is defined by a single slope parameter, the scalar σ, that can be determined during encoding by a single parameter search minimizing an RD-based cost.

C. TRIMMING

As will be detailed in Subsection IV-E, in the Slant Tree Optimization module, a 4D block is recursively partitioned and slanted. That is, given a 4D block to code, it is first slanted, then the resulting slanted 4D block is split and each resulting 4D sub-blocks are recursively slanted and split or not, always following an RD criterion. This process is guided by a Slant tree, with the iterative search performed so that the tree leaves contain slanted 4D blocks such that the 4D-DCT applied to their padded versions result in better compression performance. This implies that the final slanted 4D sub-blocks at the leaves of this tree are the result of cascades of 4D block splits and Slant transforms.

In the remainder of this paper, $\boldsymbol{\pi}_{k\times m}$ represents a 2D plane in 4D space that is parallel to the $k \times m$ axes. Likewise, $\boldsymbol{\pi}_{l\times n}$ represents a 2D plane in 4D space that is parallel to the $l \times n$ axes.

Fig. 7 illustrates the effect, on a 2D block on a plane $\boldsymbol{\pi}_{k\times m}$ (Fig. 7a), of a 2D Slant transform with slope parameter σ (Fig. 7b), followed by a split of the slanted 2D block (Fig. 7c), with the two resulting halves being applied 2D Slant transforms with slope parameters σ' and σ'' (Fig. 7d). As can be observed in the right-hand side of Fig. 7d, this might lead to a 2D block where all samples are empty above or below some spatial coordinate m at all views k (in this specific case, samples (k, m) are empty for $m''_{RB} < m \leq m''_{RT}$ for all k). Therefore, these samples can be safely trimmed before the padding operation is performed, without any data loss, thus reducing the dimensions of the block to be encoded, which tends to save bits during encoding.

The trimming procedure is detailed in Algorithm 2.

Algorithm 2 Trimming of a Slanted 4D Block

1. *Inputs*:
   - *Slanted 4D block:* $B^{\sigma_1,\ldots,\sigma_r;l_b,k_b}$, as generated by the sequential application of 4D block splits and Algorithm 1 with slope parameters $\sigma_1,\ldots,\sigma_r$; it is contained in the region $l \in \mathbb{N} \cap [0, L-1]$, $k \in \mathbb{N} \cap [0, K-1]$, $n \in \mathbb{N} \cap [n'_L, n'_R]$ and $m \in \mathbb{N} \cap [m'_L, m'_R]$.
2. *Output:*
   - *Trimmed slanted 4D block:* $B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}$.
3. *Initialization:*
   - $B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}(l, k, n, m) =$ empty, for $l \in \mathbb{N} \cap [0, L-1]$, $k \in \mathbb{N} \cap [0, K-1]$, $n \in \mathbb{N} \cap [n'_L, n'_R]$ and $m \in \mathbb{N} \cap [m'_L, m'_R]$.
   - *List of spatial coordinates* (n, m) *of empty samples:* $\mathcal{C} =$ empty.
4. *Generation of the trimmed slanted 4D block:*
   for $n \in \mathbb{N} \cap [n'_L, n'_R]$
   for $m \in \mathbb{N} \cap [m'_L, m'_R]$
   if $B^{\sigma_1,\ldots,\sigma_r;l_b,k_b}(l, k, n, m) =$ empty, $\forall\, k, l$ then
   add coordinate (n, m) to $\mathcal{C}$.
   end
   end
   · Determine $n_L^{\mathcal{T}}$, $n_R^{\mathcal{T}}$, $m_L^{\mathcal{T}}$ and $m_R^{\mathcal{T}}$ such that $(n, m) \in \mathcal{C}$ for $n \in \mathbb{N} \cap [n_L^{\mathcal{T}}, n_R^{\mathcal{T}}]$ and $m \in \mathbb{N} \cap [m_L^{\mathcal{T}}, m_R^{\mathcal{T}}]$.
   · Build $B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}$ with the coordinates (l, k, n, m) from $B^{\sigma_1,\ldots,\sigma_r;l_b,k_b}$ such that
   · $n \in \mathbb{N} \cap \left\{[n'_L, n'_R] - [n_L^{\mathcal{T}}, n_R^{\mathcal{T}}]\right\}$,
   · $m \in \mathbb{N} \cap \left\{[m'_L, m'_R] - [m_L^{\mathcal{T}}, m_R^{\mathcal{T}}]\right\}$,
   · $l \in \mathbb{N} \cap [0, L-1]$ and $k \in \mathbb{N} \cap [0, K-1]$.

Note that for the intersection of the 4D block with a plane $\boldsymbol{\pi}_{k\times m}$ shown in the right hand side of Figs. 7d and 7e, in Algorithm 2, one should make the input variables

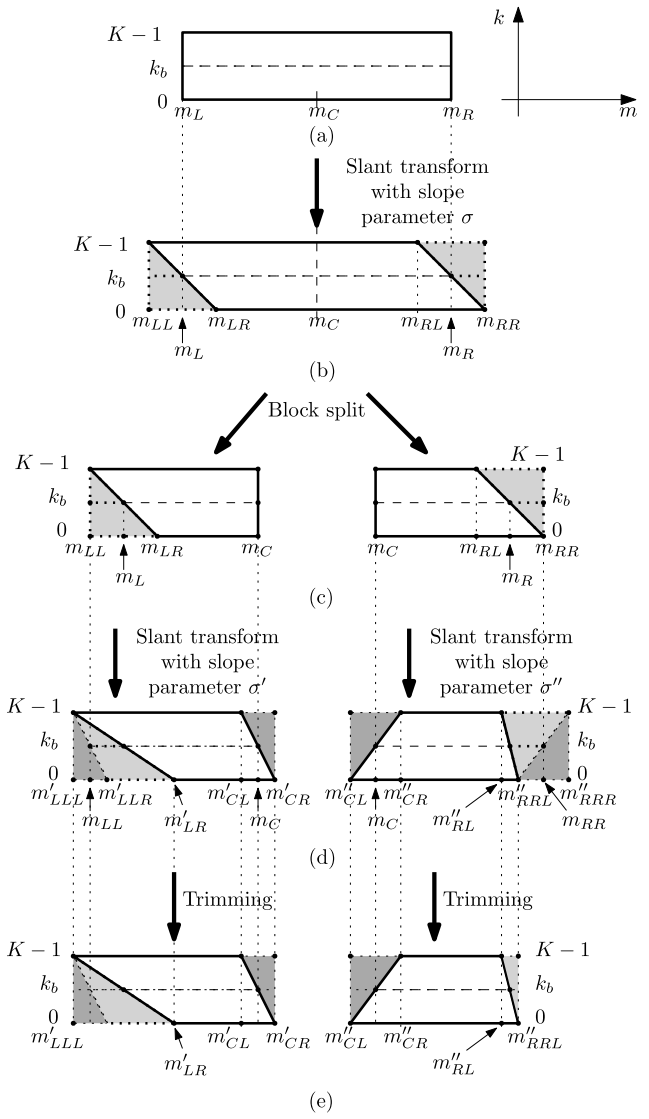


FIGURE 7. Slant-split-trim sequence illustrated on a plane $\pi_{k\times m}$. (a) Original 2D block. (b) 2D block in (a) after being applied a Slant transform with slope parameter σ. (c) Slanted block in (b) after splitting. (d) Slant transforms of the blocks in (c) with slope parameters σ' and σ'', $\sigma' \neq \sigma''$. The samples to be trimmed are on the right-hand side of (d), for $m''_{RRL}+1 \leq m \leq m''_{RRR}$. Note that in this example there is no need for trimming on the left-hand side. (e) Blocks after the trimming operation. Note that, after trimming, the non-empty samples define trapezoids.

$m'_L = m''_{CL}$, $m'_R = m''_{RRL}$, and the m coordinates to be trimmed should be $m^{\mathcal{T}}_L = m''_{RRL}+1$ and $m^{\mathcal{T}}_R = m''_{RRR}$.

Note that, from Fig. 7 and Algorithm 2, the output of the Trimming module, when intercepted by planes $\boldsymbol{\pi}_{k\times m}$ or $\boldsymbol{\pi}_{l\times n}$, yields trapezoids. The 4D output of the Trimming module can then be referred to as a 4D hyper-trapezoid, which is the result of trimming the hyper-rectangular regions of empty samples in the 4D sheared hyper-parallelepipeds.

D. PADDING

Subsection IV-B has presented in detail the design of the adopted 4D Slant transform, which has the goal of maximizing the 4D-DCT compression efficiency for a given

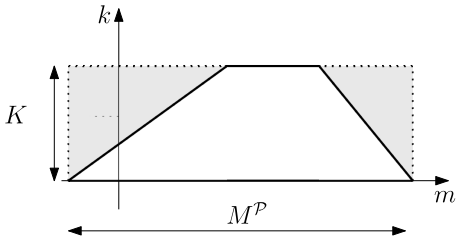


FIGURE 8. The white trapezoid is the intersection of the trimmed slanted 4D block with a plane $\pi_{k\times m}$. The dotted rectangle with dimensions $K \times M^{\mathcal{P}}$ is the intersection of $\pi_{k\times m}$ with the 4D hyper-rectangle to which the 4D hyper-trapezoid is inscribed. The gray area corresponds to the coordinates of the empty samples.

4D block. It has also been shown that the Slant transform of an input $L \times K \times N \times M$ 4D block is a sheared hyper-parallelepiped. This 4D sheared hyper-parallelepiped is then recursively partitioned, slanted, and trimmed, as detailed in Subsections IV-B, IV-C and IV-E, generating non-empty samples that are finally organized as a 4D hyper-trapezoid. However, since the 4D-DCT operation in the 4D-TM codec needs as input a hyper-rectangular 4D block, it is essential that this 4D hyper-trapezoid is appropriately completed to become a 4D hyper-rectangle.

Assuming that, in the 4D block input to the Slant Transform module, the view indexes l and k are such that $0 \leq l \leq L-1$ and $0 \leq k \leq K-1$, then from Fig. 7 and Algorithm 2, the sheared 4D hyper-trapezoid output by the Trimming module can be inscribed into a 4D hyper-rectangle of dimensions $L \times K \times N^{\mathcal{P}} \times M^{\mathcal{P}}$. This is illustrated in Fig. 8, where it is shown the 2D trapezoid that is the intersection of the discrete 4D hyper-trapezoid with a plane $\boldsymbol{\pi}_{k\times m}$. The dotted $K \times M^{\mathcal{P}}$ rectangle is the intersection of the 4D hyper-rectangle circumscribed to the 4D hyper-trapezoid with $\boldsymbol{\pi}_{k\times m}$. Note that, from Algorithm 1, Algorithm 2, Eq. (11) and Fig. 7, this 4D hyper-rectangle includes empty samples, corresponding in Fig. 8 to the gray regions. These empty samples, and the 2D trapezoid consisting of the non-empty samples belonging to a plane $\boldsymbol{\pi}_{k\times m}$ that has the coordinates $(l, n) = (0, 644)$, can also be observed in Fig. 5(c).

From the above, it is possible to generate the full 4D block required for the application of the 4D-DCT by assigning color values to the empty samples of the $L \times K \times N^{\mathcal{P}} \times M^{\mathcal{P}}$ 4D hyper-rectangle circumscribed to the 4D hyper-trapezoid. This is the key goal of the Padding module (Fig.4), which the algorithm is described in this subsection. A fundamental target is that such value assignment is performed in a way that maximizes the 4D-DCT compression efficiency.

The Padding operation is implemented in two major steps:

Step 1: Identifying the coordinates (l, k, n, m) of the empty samples in the 4D block $B^{\sigma_1,\ldots,\sigma_r;l_b,k_b}_{\mathcal{T}}$ generated by a series of 4D block splits, followed by the Slant transform and trimming.

Step 2: Assigning color values to the identified empty samples so that the 4D-DCT compression efficiency for the

4D hyper-parallelepiped circumscribed to $B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}$ is maximized.

a: STEP 1

The coordinates of the empty samples to be padded can be determined by analyzing the 2D trapezoids that are the intersections of the non-empty samples of the slanted and trimmed 4D block $B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}$ with planes $\boldsymbol{\pi}_{k\times m}$ and $\boldsymbol{\pi}_{l\times n}$. For each of these planes, four types of trapezoids are possible, as illustrated in Figs. 9a to 9d for a plane $\boldsymbol{\pi}_{k\times m}$. Note that there are different relative values of m_L, m_{LB}, m_{LT}, m_R, m_{RB} and m_{RT} for each of the four types. For the trapezoids in the intersection with a plane $\boldsymbol{\pi}_{l\times n}$, K should be replaced by L, k_b by l_b and m_X by n_X in Fig. 9. The gray areas correspond to the location of the empty samples that need padding, i.e., that should be assigned values.

b: STEP 2

Once the coordinates of the empty samples are determined (details in Algorithm 3), then Step IV-D, where each empty sample is assigned an appropriate value, may be performed by taking into account that, for a Lambertian light field in the absence of occlusions, the image of a 3D-space point P with depth z in the light field has a single color value $c(P)$. Therefore, as pointed out in Subsection IV-B, if a proper sequence of Slant transforms is applied, the image of P in the slanted and trimmed 4D block $B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}$ has the same (n_0, m_0) coordinates for all view indexes (l, k) in which it is visible, all these with constant color $c(P)$. As illustrated in Fig. 6 and discussed in Subsection IV-B, if a 3D-space point P_2 is not visible in a view (l_{nv}, k_{nv}) from the original light field, then in $B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}$ the corresponding sample $(l_{nv}, k_{nv}, n_0, m_0)$ will be empty. From the above, a natural way to perform padding is to assign to each empty sample the color value $c(P)$ present in the samples (l_v, k_v, n_0, m_0) of the corresponding visible views, that is,

$$B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}(l_{nv}, k_{nv}, n_0, m_0) = B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}(l_v, k_v, n_0, m_0) = c(P), \tag{13}$$

for all views (l_{nv}, k_{nv}) for which the sample $B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}(l_{nv}, k_{nv}, n_0, m_0)$ of the slanted and trimmed 4D block is empty. This implies that after padding

$$B_{\mathcal{P}}(l, k, n_0, m_0) = c(P),\ 0 \le l \le L-1,\ 0 \le k \le K-1. \tag{14}$$

Therefore, given the coordinates (n_0, m_0) of the 3D-space point P, the 4D slanted and trimmed block will be such that $B_{\mathcal{P}}(l, k, n_0, m_0)$ is a rectangular 2D block along the l and k directions with constant color $c(P)$. Then, its 2D-DCT will have only the DC coefficient as non-zero. Since the 4D-DCT is separable, that is, it is equivalent to a 2D-DCT applied to the l and k directions followed by a 2D-DCT in the n and m directions of the result, if the imaged 3D-space points have depths approximately equal to the one of P, the 4D-DCT applied to the slanted, trimmed and padded 4D block will have excellent energy compaction properties.

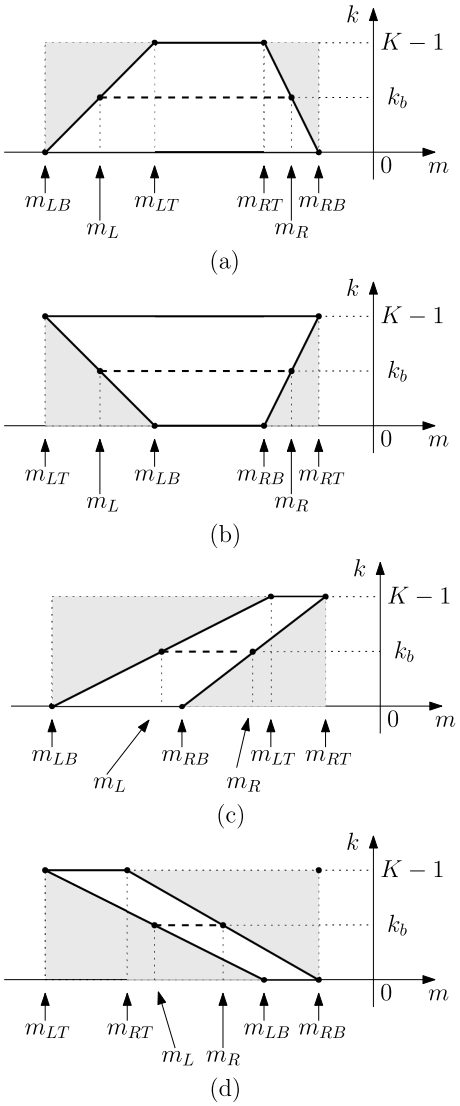


FIGURE 9. Illustration of the four possible 2D trapezoid types resulting from the intersection, with a plane $\pi_{k\times m}$, of the non-empty samples of the 4D block $B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}$ resulting from a sequence of splits, Slant transforms and trimming operations. (a), (b), (c), and (d) represent the four possible types, depending on the relative values of m_L, m_{LB}, m_{LT}, m_R, m_{RB} and m_{RT}. For the intersection with a plane $\pi_{l\times n}$, K should be replaced by L, k_b by l_b and m_X by n_X.

To describe more precisely Step 2 of the padding algorithm outlined above, it is convenient to refer to Fig. 10, that illustrates the padding operation on a plane $\boldsymbol{\pi}_{k\times m}$. The empty samples are those in the gray regions. Fig. 10 shows the values k_U and k_D such that the empty samples (k, m) for $k < k_D$ should be assigned the value of sample (k_D, m). Likewise, the empty samples (k, m) for $k > k_U$ should be assigned the value of sample (k_U, m). For the four possible trapezoids illustrated in Fig. 9, Eq. (15) defines the intervals I_1^m, I_2^m, I_3^m, I_1^n, I_2^n and I_3^n used to compute k_U, k_D, l_U and l_D in Table 1. For a plane $\boldsymbol{\pi}_{l\times n}$, one should replace, in Fig. 10, Table 1 and Eq. (15), k

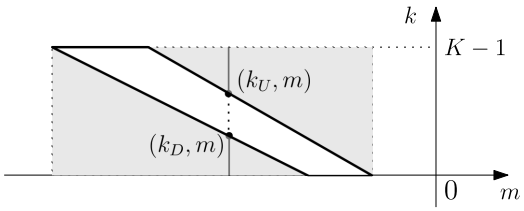


FIGURE 10. Illustration of empty samples to be padded in a plane $\pi_{k\times m}$. The empty samples are those in the gray regions. The expressions for k_D and k_U are given according to Fig. 9, Eq. 15 and Table 1.

TABLE 1. Expressions for the computation of k_U and k_D for each of the four types of trapezoids (a), (b), (c) and (d) defined in Fig. 9 for m in the intervals defined in Eq. (15), where $k_L = \left\lfloor (K-1)\frac{m-m_{LB}}{m_{LT}-m_{LB}} \right\rceil$ and $k_R = \left\lfloor (K-1)\frac{m-m_{RB}}{m_{RT}-m_{RB}} \right\rceil$. To obtain l_U and l_D, the limits for a plane $\pi_{l\times n}$, m should be replaced by n, I_i^m by I_i^n, m_X by n_X, K by L, k_U by l_U, k_D by l_D, k_L by l_L, and k_R by l_R.

m	I_1^m	I_2^m	I_3^m
k_U	k_L	$K-1$	k_R
k_D	0	0	0

(a)

m	I_1^m	I_2^m	I_3^m
k_U	$K-1$	$K-1$	$K-1$
k_D	k_L	0	k_R

(b)

m	I_1^m	I_2^m	I_3^m
k_U	$\min\{K-1, k_L\}$	$\min\{K-1, k_L\}$	$\min\{K-1, k_L\}$
k_D	$\max\{0, k_R\}$	$\max\{0, k_R\}$	$\max\{0, k_R\}$

(c)

m	I_1^m	I_2^m	I_3^m
k_U	$\min\{K-1, k_R\}$	$\min\{K-1, k_R\}$	$\min\{K-1, k_R\}$
k_D	$\max\{0, k_L\}$	$\max\{0, k_L\}$	$\max\{0, k_L\}$

(d)

by l, m by n, K by L, k_U by l_U and k_D by l_D.

$$
\begin{aligned}
&\text{Trapezoid (a):} \quad m_{LB} \le m_{LT} \le m_{RT} \le m_{RB},\\
&I_1^m = [m_{LB}, m_{LT}],\ I_2^m = [m_{LT}, m_{RT}],\ I_3^m = [m_{RT}, m_{RB}],\\
&\text{Trapezoid (b):} \quad m_{LT} \le m_{LB} \le m_{RB} \le m_{RT},\\
&I_1^m = [m_{LT}, m_{LB}],\ I_2^m = [m_{LB}, m_{RB}],\ I_3^m = [m_{RB}, m_{RT}],\\
&\text{Trapezoid (c):} \quad m_{LB} \le m_{LT},\ m_{RB} \le m_{RT},\ m_{LB} \le m_{RB},\\
&I_1^m = I_2^m = I_3^m = [m_{LB}, m_{RT}],\\
&\text{Trapezoid (d):} \quad m_{LT} \le m_{LB},\ m_{RT} \le m_{RB},\ m_{LT} \le m_{RT},\\
&I_1^m = I_2^m = I_3^m = [m_{LT}, m_{RB}]. \qquad (15)
\end{aligned}
$$

For a plane $\pi_{k\times m}$, I_i^m should be replaced by I_i^n and m_X by n_X.

Fig. 11 further illustrates, on the $l \times k$ plane, the padding operation performed on the empty samples of the slanted and trimmed 4D block corresponding to fixed coordinates (n, m) within each view of a slanted 4D block. The gray areas correspond to the (l, k) view coordinates of the empty samples. Note that the color value to be used to perform padding of an empty sample varies depending on the region of the $l \times k$ plane that its l and k coordinates are in. These regions are indicated in Fig. 11 by the letters A to H. The arrows indicate examples of which non-empty samples (black dots) have their values copied to perform padding of empty samples (non-filled-in circles), which depends on the region of the $l \times k$ plane, i.e., A to H, its coordinates belong to. The coordinates (n, m) should be varied so that all the empty samples to be padded are scanned, according to Fig. 9. The overall padding procedure is detailed in Algorithm 3.

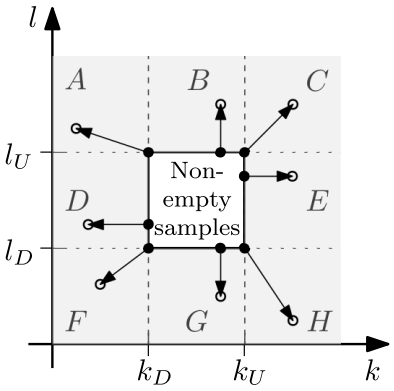


FIGURE 11. Illustration of the padding procedure for the empty samples of a slanted and trimmed 4D block in the $l \times k$ plane. The gray areas correspond to the empty samples. The eight different cases for choosing the non-empty samples that are copied in the padding operation are indicated by the regions corresponding to letters A to F. The arrows indicate the non-empty samples (black dots) whose color values are copied to perform the padding of the empty samples (non-filled-in circles).

Algorithm 3 Padding of a Slanted and Trimmed 4D Block

1. *Inputs*:
   - *Slanted and trimmed 4D block:* $B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}$, with parameters m_{LT}, m_{LB}, m_{RT}, m_{RB}, n_{LT}, n_{LB}, n_{RT} and n_{RB}, belonging to one of the four possible types (a), (b), c) and (d) defined in Fig. 9.
2. *Output:*
   - *Padded 4D block:* $B_{\mathcal{P}}$.
3. *Initialization:*
   - Intervals I_1^n, I_2^n, I_3^n, I_1^m, I_2^m and I_3^m are initialized using Eq. (15).
4. *Generation of the padded 4D block:*

```
for n ∈ ℕ ∩ {I_1^n ∪ I_2^n ∪ I_3^n}
  for m ∈ ℕ ∩ {I_1^m ∪ I_2^m ∪ I_3^m}
    k_U, k_D, l_U and l_D are computed according
      to Table 1.
    for l ∈ ℕ ∩ {[0, l_D] ∪ [l_U, L − 1]}
      for k ∈ ℕ ∩ {[0, k_D] ∪ [k_U, K − 1]}
        if k < k_D and l_D ≤ l ≤ l_U then
          B_P(l, k, n, m) = B_T^{σ_1,...,σ_r;l_b,k_b}(l, k_D, n, m)
        if k > k_U and l_D ≤ l ≤ l_U then
          B_P(l, k, n, m) = B_T^{σ_1,...,σ_r;l_b,k_b}(l, k_U, n, m)
        if k_D ≤ k ≤ k_U and l < l_D then
          B_P(l, k, n, m) = B_T^{σ_1,...,σ_r;l_b,k_b}(l_D, k, n, m)
        if k_D ≤ k ≤ k_U and l > l_U then
          B_P(l, k, n, m) = B_T^{σ_1,...,σ_r;l_b,k_b}(l_U, k, n, m)
        if k < k_D and l < l_D then
          B_P(l, k, n, m) = B_T^{σ_1,...,σ_r;l_b,k_b}(l_D, k_D, n, m)
        if k > k_U and l < l_D then
```

$B_{\mathcal{P}}(l,k,n,m) = B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}(l_D,k_U,n,m)$
if $k < k_D$ and $l > l_U$ then
$B_{\mathcal{P}}(l,k,n,m) = B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}(l_U,k_D,n,m)$
if $k > k_U$ and $l > l_U$ then
$B_{\mathcal{P}}(l,k,n,m) = B_{\mathcal{T}}^{\sigma_1,\ldots,\sigma_r;l_b,k_b}(l_U,k_U,n,m)$
end
end
end
end

The padding performed by Algorithm 3 (see an example of a slanted, trimmed, and padded EPI in Fig. 5d) provides maximum compression if applied to a properly slanted 4D block that is the image of a region having a single depth. When this condition is not met, the best slope parameter σ such that the 4D-DCT compression performance of the slanted, trimmed and padded 4D block is maximum should be searched for. An efficient way to achieve this target is proposed in the next subsection describing the Slant Tree Optimization module.

E. SLANT TREE OPTIMIZATION

As outlined in Subsection IV-A, the goal of the Slant Tree Optimization module is to find the optimal partition tree for each 4D block to be coded, along with the optimal Slant transform slope parameter for each 4D sub-block resulting from the partition. The need for 4D block partitioning arises because, when performing the Slant transform of a large 4D block, the different slopes that are likely to be present in such a large 4D block will be less effectively compensated when compared to what can be achieved with smaller 4D blocks. Referring to Fig. 8, the larger the view dimensions N and M of the original 4D block to be slanted, the smaller the overhead due to padding, which is given by $(N^{\mathcal{P}}M^{\mathcal{P}})/(NM)$. However, as mentioned above, the smaller the 4D block, the more efficient tends to be the 4D-DCT applied to the slanted, trimmed, and padded 4D block. Therefore, the choice of the initial 4D block size is driven by an RD trade-off, thus implying that there is a partition of a given 4D block that is optimal in the RD sense.

The goal of the Slant Tree Optimization module is to recursively generate a block partition tree with associated Slant transforms for its nodes that generate the optimal 4D-DCT compression efficiency. The Slant Tree Optimization module architecture is depicted in Fig. 12 and works as follows:

- *4D Block Partitioning* - Starting with the $L \times K \times N \times M$ 4D blocks originated from the Regular 4D Block Partitioning, recursively partitions these 4D blocks given a quaternary partition tree with nodes defined by binary partition flags (*Split / noSplit*).
- *Slant Transform and Trimming* - These two modules are described in detail in Subsections IV-B and IV-C which perform the slant and trimming operations on the sub-blocks resulting from the 4D block partitioning module.
- *Padding, 4D-DCT, Quantization, Inverse 4D-DCT, Inverse Slant Transform, and Padding Removal* - This sequence of modules generates an estimate of the slanted and trimmed 4D sub-block after it is recovered from the quantized 4D-DCT coefficients of its padded version. The inverse of a Slant transform with slope parameter σ is another Slant transform with the same base view and slope parameter $\sigma' = -\sigma$. Note that to compute the distortion between the original and recovered 4D sub-blocks, after the inverse Slant transform, the padded samples should be removed.
- *RD-oriented Partition and Slant Transform Slope Search* - The purpose of this sub-module is twofold. First, for a given 4D sub-block, it finds the best Slant transform in an RD sense by minimizing an RD cost. Second, it finds the 4D block partition that gives the minimum RD cost for the $L \times K \times N \times M$ 4D blocks resulting from the regular block partitioning; the outcome is a sequence of partition flags and Slant transform slope parameters that build an optimal Slant tree.

The RD-optimal Slant tree search is detailed in Algorithm 4.

Algorithm 4 RD-Optimal Slant Transform Tree Search

1. *Inputs*:
   - *4D Block*: B, contained in the region $l \in \mathbb{N} \cap [0, L-1]$, $k \in \mathbb{N} \cap [0, K-1]$, $n \in \mathbb{N} \cap [n_L, n_R]$ and $m \in \mathbb{N} \cap [m_L, m_R]$.
   - Slant transform slope parameter search range $[\sigma_{\min}, \sigma_{\max}]$.
   - Base view indexes (l_b, k_b).
   - Minimum sizes for the vertical and horizontal view dimensions that still allow a 4D block to be partitioned, namely $N_{\min}$ and $M_{\min}$, respectively.
   - Lagrangian multiplier λ to be used in the computation of the RD costs in Step 4..i..
2. *Outputs*:
   - *String of Partition flags*: *flagString*, each flag set to either *Split* or *noSplit* to signal if a block is either split or not split, respectively.
   - Optimal Slant transform slope parameter $\sigma_{\min}$ for B.
3. *Initialization*:
   - $n_d = n_R - n_L + 1$ and $m_d = m_R - m_L + 1$, the vertical and horizontal block dimensions, respectively.
   - RD costs J_{Split} and $J_{noSplit}$ and $J_{\min}$ are set to infinity.
   - Set *flagString* to the *empty* string.
4. *4D Slant transform slope parameter and 4D block partitioning search*:
   i. For $\sigma \in [\sigma_{\min}, \sigma_{\max}]$ (see Eq. (16) and the related discussion),
      - Compute $B^{\sigma;l_b,k_b}$, the Slant transform of B using Algorithm 1.
      - Compute the padded slanted 4D block $B_{\mathcal{P}}$ applying Algorithm 3 to $B^{\sigma;l_b,k_b}$.

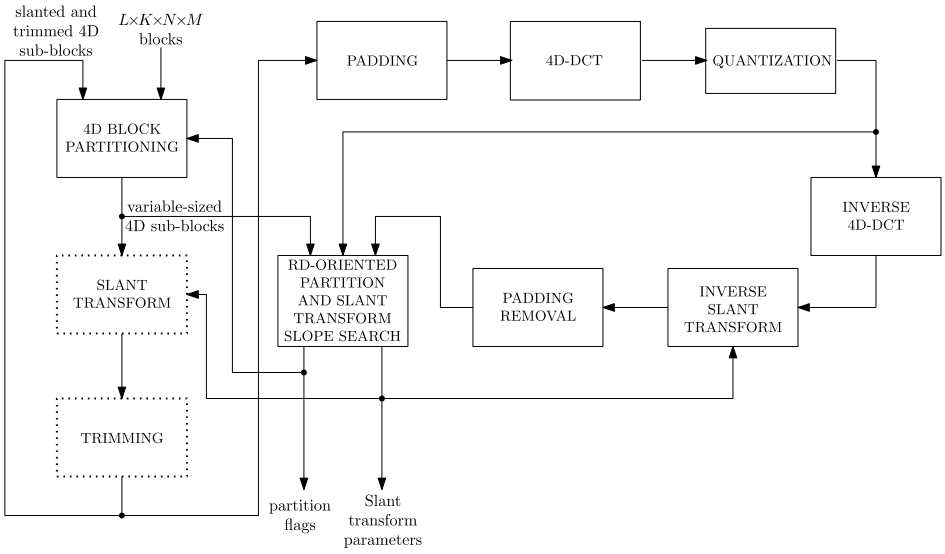


FIGURE 12. Architecture of the slant tree optimization module.

- Compute the 4D-DCT of $B_{\mathcal{P}}$, quantize its coefficients, estimate the rate R for encoding them and compute the inverse 4D-DCT of the quantized 4D-DCT coefficients, $\overline{B}_{\mathcal{P}}$.
- Obtain $\overline{B}^{\sigma;l_b,k_b}$ by removing the padded samples from $\overline{B}_{\mathcal{P}}$.
- Compute the RD cost $J = D + \lambda R$, where D is the mean squared error between $B^{\sigma;l_b,k_b}$ and $\overline{B}^{\sigma;l_b,k_b}$, and R is the entropy of the quantized 4D-DCT coefficients of $B_{\mathcal{P}}$ obtained above.
- if $J < J_{\min}$ then $J_{\min} = J$; $\sigma_{\min} = \sigma$.

end

ii. $J_{noSplit} = J$;

iii. If $n \leq N_{\min}$ and $m \leq M_{\min}$ then *Partition flag* = *noSplit*;

flagString becomes the unitary string containing *Partition flag*;

The RD cost J and the partition string *flagString* are returned and the algorithm terminates.

iv. The $L \times K \times n_d \times m_d$ 4D block B is partitioned into four sub-blocks by splitting it along its view dimensions (n, m), generating four 4D sub-blocks of dimensions
- $L \times K \times \lfloor n_d/2 \rfloor \times \lfloor m_d/2 \rfloor$,
- $L \times K \times \lfloor n_d/2 \rfloor \times (m - \lfloor m_d/2 \rfloor)$,
- $L \times K \times (n_d - \lfloor n_d/2 \rfloor) \times \lfloor m_d/2 \rfloor$ and
- $L \times K \times (n_d - \lfloor n_d/2 \rfloor) \times (m_d - \lfloor m_d/2 \rfloor)$.

Then Algorithm 4 is recursively called for each of the four sub-blocks, and the RD costs returned by these four calls are added to compute the RD cost J_{Split};

v. If $J_{noSplit} \leq J_{Split}$, then $J = J_{noSplit}$; *Partition flag* = *noSplit*.

vi. If $J_{Split} < J_{noSplit}$ then $J = J_{Split}$; *Partition flag* = *Split*.

vii. *flagString* becomes the concatenation of *Partition flag* followed by the four partition strings returned from the recursive calls.

viii. The RD cost J and the partition string *flagString* are returned and the procedure terminates.

It is important to note that the Slant transform of the 4D sub-blocks resulting from the optimal partition in Algorithm 4 is recursive, thus following an incremental strategy. A 4D block is slanted before it is partitioned into 4D sub-blocks that will be further recursively slanted, and thus the effective slant of a 4D sub-block results from the accumulation of its slant with the ones of its ascendant 4D sub-blocks. As specified in Step 4..iii., this strategy can be recursively applied until a user-defined minimum block size is reached.

Another relevant point in the implementation of the Slanted 4D-TM codec is that, in Step 4..i. of Algorithm 4, the Slant transform slope parameter is varied in the range $[\sigma_{\min}, \sigma_{\max}]$. In the results shown in Section V, this variation is performed such that the absolute value of the horizontal displacement of the 4D block view that is most displaced by the Slant transform, that is

$$\mathcal{D}_m = \max\left\{\left\lfloor \frac{\Delta s}{\Delta u}\sigma k_b \right\rceil, \left\lfloor \frac{\Delta s}{\Delta u}\sigma (K-1-k_b) \right\rceil\right\} \quad (16)$$

assumes integer values. Regarding the values of $\sigma_{\min}$ and $\sigma_{\max}$, a safe choice comes from the fact that, if the Slant transform causes a displacement among adjacent views that is larger than the view dimensions, that is, if

$$\left|\left\lfloor \frac{\Delta s}{\Delta u}\sigma \right\rceil\right| > M \quad \text{or} \quad \left|\left\lfloor \frac{\Delta t}{\Delta v}\sigma \right\rceil\right| > N, \quad (17)$$

then the views of the slanted 4D block would not share any common (n, m) coordinates. In this case, there would no redundancy in the original 4D sub-block able to be

exploited by the application of the DCT to the l and k directions of the slanted 4D sub-block. Therefore, a Slant transform with such a slope parameter would be inefficient from an RD viewpoint. Therefore, $\sigma_{\min}$ and $\sigma_{\max}$ should be chosen so that neither of the conditions in Eq. (17) happen.

From the analysis of Algorithm 4, it can be inferred that, if no Slant transform slope parameter search is performed, the complexity of the Slanted 4D-TM codec is roughly equivalent to the one of a direct and an inverse 4D-DCT. For each Slant transform slope parameter σ evaluated by Algorithm 4, an extra cascade of direct and inverse 4D-DCT is performed. This implies that the complexity of Algorithm 4 tends to increase linearly with the number of Slant transform slopes σ evaluated in its Step 4..i.. During the initial experiments, it has been verified that, in general, for the Slant transform slope search performed for the initial 4D block, that is, when it has not been partitioned yet, the Slant transform slope parameters $\sigma_{\min}$ and $\sigma_{\max}$ should be the largest ones so that the conditions in Eq. (17) are not violated. For the subsequent steps, it has been verified that there is no improvement in RD performance if the number of Slant transform slopes searched for is larger than 65 symmetric values, that is, it suffices that the σ search is performed such that $\mathcal{D}_m$ in Eq. (16) is smaller than 32. Therefore, to limit the computational complexity in the experiments reported in Section V, 65 symmetric values of σ are used such that $\mathcal{D}_m$ is an integer smaller than 32. The exception is the *Laboratory1* light field, where, to avoid excessive computational complexity, $\sigma_{\min}$ and $\sigma_{\max}$ are set such that $\mathcal{D}_m$ is an integer smaller than or equal to 10.

V. PERFORMANCE ASSESSMENT

The performance assessment of the newly proposed light field coding mode has been conducted as much as possible according to the JPEG Pleno Light Field Common Test Conditions [8]; these conditions are well recognized, representative and meaningful, thus offering credibility to the obtained results and conclusions.

A. TEST MATERIAL

To create strong evidence to support the claims made about the performance of the proposed coding mode, it is critical to select a set of test light fields that covers a wide range of spatial-angular redundancy also associated with a wide range of baselines. When coding 4D light field data, an efficient codec should exploit its spatial-angular (or space-view) redundancy. Therefore, the target is to have enough content variety to demonstrate the potential of the newly proposed coding mode fully. To characterize the light fields and obtain this variety, a quantitative descriptor, the so-called Geometric Space-View Redundancy (*GSVR*) descriptor, has been used [8], [49] since it offers a good understanding of the light field 4D scene geometry. In this context, the *GSVR* descriptor characterizes the various trade-offs involved in effectively exploring the 4D space-view redundancy (joint intra- and inter-view redundancies) of a light field for efficient coding. As such, it defines different classes of space-view redundancy which will be instrumental to show that the proposed Slanted 4D-TM codec is effectively efficient across these classes.

With the *GSVR* descriptor, one takes the average percentage of the number of views from a given 4D block that are intercepted by the image of a given point in 3D space. The space-view redundancy is measured by how this percentage varies with the dimensions of the 4D block. A small variation of this percentage with the 4D block dimensions means that the image of a 3D space point traces a long trajectory inside it, thus generating many redundant points inside the 4D block. This is equivalent to stating that there is significant 4D space-view redundancy to be exploited. The *GSVR* descriptor offers a relation between the intra-view and inter-view block dimensions, thus guaranteeing the existence of such 4D space-view redundancy with a given probability (p). As such, *GSVR(p)* is defined as the ratio of the required intra-view block size given an inter-view block size to guarantee a specific permanence probability.

In this paper, the test light fields are those recently selected for the JPEG Pleno Light Field Quality Assessment Common Test Conditions [9] developed in the context of the JPEG Pleno Light Field Quality Assessment Call for Contributions [10]. This call regards the activity targeting the development of subjective quality assessment protocols and objective quality metrics for light fields. Example views of each test light field are pictured in Fig. 13.

Fig. 14 pictures the *GSVR* curves for the selected light fields. These light fields [9], [50] show a large variety of *GSVR* behaviors since they consider different scene geometries, spatial resolutions, content type, and light field acquisition/creation technologies, thus highlighting that diverse test material is being selected. The *GSVR* values in Fig. 14 show that the selected light fields may be clustered into three main classes. As expected, the four lenslet light fields show the lowest *GSVR* values (dashed lines), indicating that they have the highest space-view redundancy and that the inter-view redundancy of their 4D blocks can be well exploited. The middle cluster includes the *Sideboard* and *Tarot* light fields which present intermediate *GSVR* values. Finally, the last cluster, including the *Greek* and *Laboratory1* light fields, offers the highest *GSVR* values and thus the lowest space-view redundancy.

With the help of the *GSVR* descriptor, the selected test light fields are now clustered in terms of their space-view redundancy since, depending on their characteristics, they offer different challenges to the various light field coding solutions. The selected test light fields are presented in Table 2 coarsely clustered by their space-view redundancy as given by the *GSVR* descriptor. The last column shows the dimensions of the light fields in terms of the number of vertical and horizontal views and their spatial resolution.

FIGURE 13. Example views for the selected test light fields: (a) ***Bikes*** and ***Danger de Mort*** (top), and ***Stone Pillars Outside*** and ***Fountain&Vincent2*** (bottom); (b) ***Sideboard*** and ***Tarot***; (c) ***Greek*** and ***Laboratory1***.

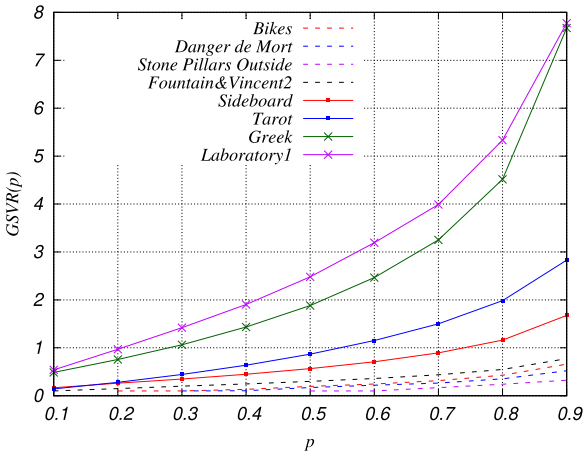


FIGURE 14. ***GSVR*** **as a function of the permanence probability for the selected test light fields.**

B. PERFORMANCE METRICS

The adopted objective quality metric is PSNR-YUV (dB), defined as in the JPEG Pleno CTC [8]:

$$\text{PSNR-YUV} = \frac{6\,\text{PSNR-Y} + \text{PSNR-U} + \text{PSNR-V}}{8} \quad (18)$$

TABLE 2. Test light fields clustered according to their space-view redundancy [50].

Space-view redundancy	Name	Dimensions ($t \times s \times v \times u$)
High	*Bikes* *Danger de Mort* *Fountain & Vincent2* *Stone Pillars Outside*	$13\times13\times434\times625$
Medium	*Sideboard* *Tarot*	$9\times9\times512\times512$ $17\times17\times1024\times1024$
Low	*Greek* *Laboratory1*	$9\times9\times512\times512$ $31\times31\times1288\times1936$

As expected, the PSNR-YUV computation gives a higher weight to the Y component and assumes a dynamic range of 10 bits. The input and output light field views are in RGB format, converted to YUV 4:4:4 prior to the PSNR-YUV computation [8]. The light field PSNR-YUV corresponds to the average of the individual PSNR-YUV for all coded views.

The PSNR-YUV RD performance for the various codecs is compared using the Bjøntegaard delta rate [51] (BD-Rate) with the piecewise cubic interpolation implementation [52].

C. CODING CONDITIONS

The light fields listed in Table 2 have been coded with the proposed Slanted 4D-TM codec according to the JPEG Pleno Common Test Conditions (CTC) [8]. The light fields have been coded with the rates specified in [8], where the rate in bpp (bits per pixel) is computed as the ratio between the total number of bits used to code the light field and the total number of samples in the light field. For the lenslets and *Tarot* light fields, the recommended rates are 0.001 bpp, 0.005 bpp, 0.02 bpp, 0.1 bpp and 0.75 bpp, while for the remaining light fields, the recommended rates are 0.0005 bpp, 0.001 bpp, 0.005 bpp, 0.01 bpp, 0.05 bpp, and 0.1 bpp. To obtain the best RD performance for the Slanted 4D-TM codec, i.e., exploiting the full redundancy as much as possible, the initial block size was set to the size of the entire light field for all of them except the *Laboratory1* light field. *Laboratory1* has an initial block size of $31 \times 31 \times 644 \times 968$, due to its huge size. The RD performance results for the standard 4D-TM codec reported in [42] have been obtained for smaller initial block sizes, which tend to reduce the RD performance. Therefore, to keep the comparison fair, the light fields have been coded, for both 4D-TM and Slanted 4D-TM codecs, using the initial block sizes defined above, which are larger than those in [42]. The RD performance results shown here for the Slanted 4D-TM have been obtained with the software implementation publicly available at [53].

D. BENCHMARKS

To assess the Slanted 4D-TM compression efficiency, its RD performance is compared with several relevant light field coding solutions in the literature. While the first three benchmarks are direct standard coding solutions, the last three

are standard-based coding solutions with some extensions, thus not standard compliant. It is critical to highlight that only the first two benchmarks are royalty-free, while the last four are heavily burdened by licensing. Licensing is a codec comparison dimension that has to be kept in mind beyond compression performance to consider the market constraints appropriately. In summary, the benchmarks are:

- **4D-TM**, which corresponds to the JPEG Pleno Light Field coding mode specified in the JPEG Pleno Light Field Coding standard [3], [46] that the proposed coding mode targets to improve; the performance results are obtained with the JPEG Pleno VM2.1 Verification Model software [54];
- **4D-PM**, which corresponds to the other JPEG Pleno Light Field coding mode specified in the JPEG Pleno Light Field Coding standard [3], [46]; the performance results are obtained with the JPEG Pleno VM2.1 Verification Model software [54];
- **HEVC**, which performs pseudo-video HEVC coding of the $s \times t$ views in an IPPP serpentine scanning pattern (left-right, top-down); this is the coding reference/anchor recommended in the JPEG Pleno Common Test Conditions (CTC) [8]. The performance results are obtained with the x265 [55] implementation [8];
- **FDL-hierarchical**, which corresponds to the light field coding method proposed in [36] based on Fourier Disparity Layer (FDL) representation using a binary hierarchical method and HEVC; this solution was reviewed in Section II-A;
- **WaSPR**, which corresponds to the light field coding method proposed in [45]; this is an improved version of the 4D-PM codec, employing HEVC instead of JPEG 2000 to code the views, residuals, and depth data; this method was reviewed in Section II-A;
- **MVI-VVC**, which corresponds to the light field coding method proposed in [24] using VVC; this method was reviewed in Section II-A.

The RD performance data reported for FDL-hierarchical have been directly obtained from the literature [36], while authors of the WaSPR [45] and the MVI-VVC [24] codecs kindly provided the results included in this paper. The comparisons have been restricted to the light field coding methods for which results using the JPEG Pleno CTC [8] are available.

E. JPEG PLENO RD PERFORMANCE ASSESSMENT

The core idea of the newly proposed Slanted 4D-DCT coding mode is to efficiently exploit the 4D redundancy of both densely and non-densely angular sampled light fields, including a large variety of scene geometries to outperform the already available JPEG Pleno light field coding modes. Hence, its RD performance must be assessed and compared for light fields presenting high, medium, and low space-view redundancy as discussed in Subsection V-A.

Table 3 shows the PSNR-YUV Bjøntegaard delta rate [51] for the proposed Slanted 4D-TM codec, for all light fields in Subsection V-A, regarding the already specified JPEG Pleno light field 4D-TM and 4D-PM codecs [3]. The results clearly show that the new Slanted 4D-TM codec is able to always outperform the 4D-TM and 4D-PM modes as promised. Naturally, the RD performance gains vary for different light field characteristics, as will be discussed in the sequel.

TABLE 3. BD-Rate (%) for the Slanted 4D-TM codec regarding the 4D-TM and 4D-PM codecs [3] for all test light fields.

	Slanted 4D-TM vs 4D-TM	Slanted 4D-TM vs 4D-PM
Bikes	−1.46	−37.67
Danger	−2.72	−34.91
Fountain	−3.37	−52.31
Pillars	−7.65	−21.38
Average	−3.80	−36.57
Sideboard	−49.96	−0.68
Tarot	−68.73	−42.54
Average	−59.35	−21.61
Greek	−63.37	−21.79
Laboratory1	−51.01	−15.11
Average	−57.19	−18.45
Total Average	−31.03	−28.30

1) RD PERFORMANCE FOR HIGH SPACE-VIEW REDUNDANCY LIGHT FIELDS

The lenslet light fields present different degrees of spatial complexity and are inherently densely angular sampled, thus offering high levels of space-view redundancy, as evidenced by the *GSVR* curves cluster with the lowest *GSVR* values in Fig. 14. The RD performance results reported in Table 3 and in Figs. 15a to 15d allow deriving the following conclusions:

a: SLANTED 4D-TM VERSUS 4D-TM

- In terms of RD performance, the proposed Slanted 4D-TM codec outperforms the 4D-TM codec for all lenslet light fields.
- As expected, the Slanted 4D-TM BD-Rate gains over the 4D-TM codec are small, with an average gain of 3.80% (Table 3).
- Figs. 15a to 15d, which show the RD performance for the lenslet light fields, make it clear that the Slant transform has a small impact on the Slanted 4D-TM performance. In this case, they report very small BD-Rate gains over the 4D-TM codec. This was expected since these light fields have the lowest *GSVR* values (Fig. 14). In addition, the available 4D-TM coding mode is already very efficient for this type of light field since their 4D blocks tend to have the σ in Eq. (1) and angle β in Fig. 6 close to zero.

b: SLANTED 4D-TM VERSUS 4D-PM

- In terms of RD performance, the proposed Slanted 4D-TM codec outperforms the 4D-PM codec for all lenslet light fields.

- The Slanted 4D-TM BD-Rate gains over the 4D-PM codec are significantly high when coding these densely angular sampled light fields, with an average gain of 36.57% (Table 3).
- Figs. 15a to 15d, which show the RD performance for the lenslet light fields, make clear that both the 4D-TM and Slanted 4D-TM codecs outperform the 4D-PM codec, thus corroborating the BD-Rate results in Table 3.

2) RD PERFORMANCE FOR MEDIUM SPACE-VIEW REDUNDANCY LIGHT FIELDS

The computer-generated *Sideboard* and the natural indoor scene *Tarot* light fields represent here the medium space-view redundancy *GSVR* cluster, as shown in Fig 14. The obtained RD performance results reported in Table 3 and Figs. 16a and 16b allow deriving the following conclusions:

a: SLANTED 4D-TM VERSUS 4D-TM

- In terms of RD performance, the proposed Slanted 4D-TM codec outperforms the 4D-TM codec for both medium space-view redundancy light fields.
- The Slanted 4D-TM BD-Rate gains over the 4D-TM codec are quite high, on average 59.35% (Table 3). This codec offers a significant improvement regarding the 4D-TM codec, thus confirming that the proposed slanting scheme is able to reshape the light field data to maximize the 4D-DCT compactness capability.
- Figs. 16a and 16b which show the RD performance for the *Sideboard* and *Tarot* light fields, respectively, make clear that the Slant transform, as targeted, has a large impact on the Slanted 4D-TM RD performance in this case. It provides, as shown in Table 3, very large BD-Rate gains over the 4D-TM codec.

b: SLANTED 4D-TM VERSUS 4D-PM

- In terms of RD performance, the proposed Slanted 4D-TM codec outperforms the 4D-PM codec for both medium space-view redundancy light fields; this is a key achievement since the 4D-PM mode was standardized due to the poor RD performance offered by the 4D-TM codec for this type of light fields.
- The Slanted 4D-TM BD-Rate gains over the 4D-PM codec (Table 3) amount to 21.61%, highlighting that the proposed coding mode is competitive to code these medium space-view redundancy light fields. Their 4D blocks tend to have σ in Eq. (1) and angle β in Fig. 6 with significantly larger values than those from the light fields classified as having high space-view redundancy, thus offering room for Slant transform-based gains.
- The computer-generated *Sideboard* light field presents a challenging scene geometry and a high degree of spatial complexity; moreover, a very accurate depth map is available, thus favoring depth data-dependent light field coding methods, i.e. the 4D-PM codec. Nevertheless, the proposed Slanted 4D-TM codec is able to outperform the depth-based 4D-PM codec.
- Figs. 16a and 16b, which show the RD performance for the *Sideboard* and *Tarot* light fields, respectively, make it clear that the Slant transform is very effective in improving the RD performance for both light fields to the point it is quite competitive with the one of the 4D-PM codec.

Despite presenting lower *GSVR* values (Fig. 14) than those for the *Tarot* light field, the BD-rate gains for *Sideboard* light field are less expressive (Table 3). This fact shows that compression efficiency depends not only on the light field redundancy but also on the overall spatial and geometrical properties of the light field.

3) RD PERFORMANCE FOR LOW SPACE-VIEW REDUNDANCY LIGHT FIELDS

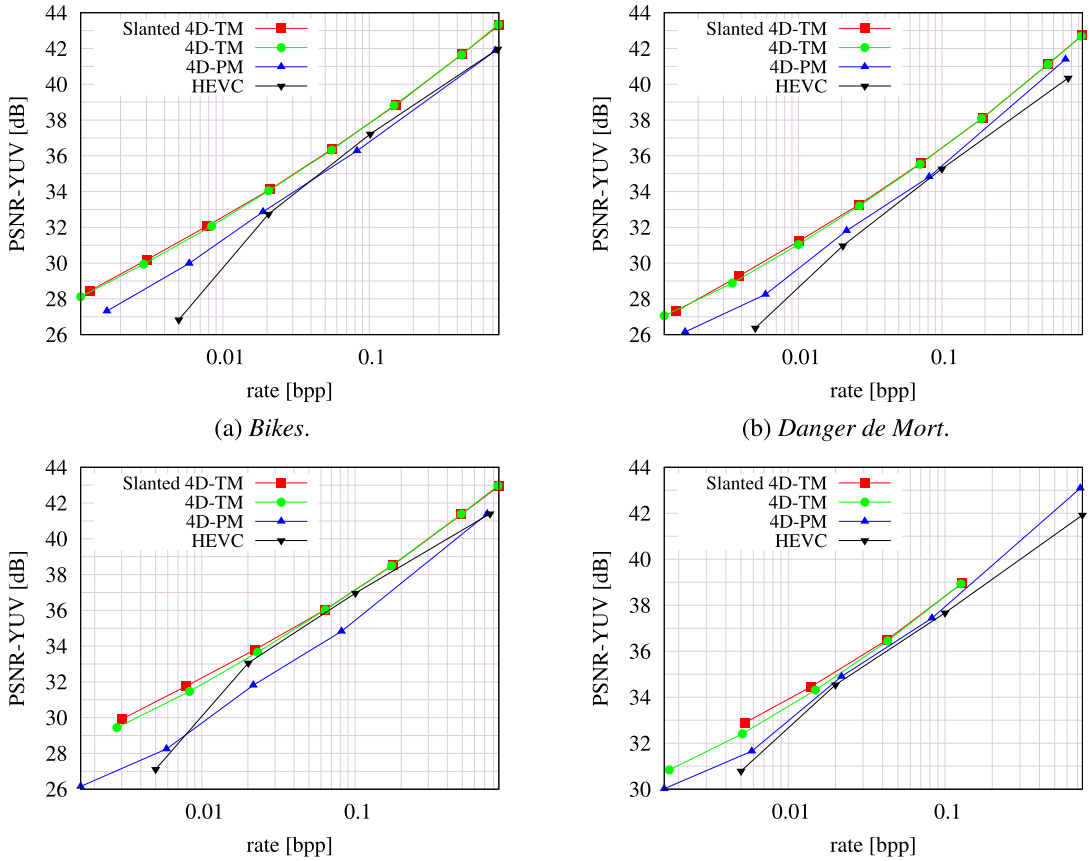
The computer-generated *Greek* and the natural indoor scene *Laboratory1* light fields present low space-view redundancy, as shown in Fig 14, thus posing a challenge to any light field coding method. The obtained RD performance results reported in Table 3, Figs. 17a and 17b lead to the following conclusions:

a: SLANTED 4D-TM VERSUS 4D-TM

- In terms of RD performance, the proposed Slanted 4D-TM codec outperforms the 4D-TM codec for both light fields; this is again a key achievement since the 4D-PM mode was standardized due to the poor RD performance offered by the 4D-TM codec for this type of light fields.
- The Slanted 4D-TM BD-Rate gains over the 4D-TM codec are quite high, on average 57.19% (Table 3). The proposed Slanted 4D-TM codec offers a large improvement regarding the 4D-TM codec, thus showing that the proposed slanting scheme is able to reshape this type of light field data to maximize the 4D-DCT compactness capability.
- Figs. 17a and 17b, which show the RD performance for the *Greek* and *Laboratory1* light fields, respectively, make it clear that the Slant transform has a great impact on the Slanted 4D-TM RD performance, in this case offering significant BD-Rate gains over the 4D-TM codec as targeted and shown in Table 3.

b: SLANTED 4D-TM VERSUS 4D-PM

- In terms of RD performance, the proposed Slanted 4D-TM codec outperforms the 4D-PM codec for both light fields.
- The Slanted 4D-TM BD-Rate gains over the 4D-PM codec are moderate, on average 18.45% (Table 3) for these low space-view redundancy light fields. Their 4D blocks tend to have σ in Eq. (1) and angle β in Fig. 6 with significantly larger values than those from the low and medium space-view light fields, confirming once again the proposed codec capability to generate geometrically suitable data for the 4D-DCT.
- Even for the synthetic *Greek* light field, which has an accurate depth map available, the novel Slanted 4D-



(a) *Bikes*. (b) *Danger de Mort*.

(c) *Fountain&Vincent2*. (d) *Stone Pillars Outside*.

FIGURE 15. RD performances for the high space-view redundancy light fields.

TM codec is able to provide a large BD-Rate gain over the depth-based 4D-PM codec, clearly showing its compression efficiency power.

- Fig. 17a and 17b, which shows the RD performance for the *Greek* and *Laboratory1* light fields, respectively, make it clear that the Slant transform is very efficient in improving the RD performance for both light fields, to the point that they consistently outperform the 4D-PM.

In summary, the RD performance results convincingly show that the proposed Slanted 4D-TM codec outperforms the already standardized JPEG Pleno Light Field 4D-TM and 4D-PM codecs. This implies that a single coding mode can efficiently code all types of light fields, thus not only avoiding the current need to select one of the two coding modes but also consistently offering RD performance gains. This conclusion seems to justify the standardization of an additional JPEG Pleno Light Field coding mode (through an amendment), thus without breaking the compatibility with the existing coding modes.

F. RD PERFORMANCE BENCHMARKING

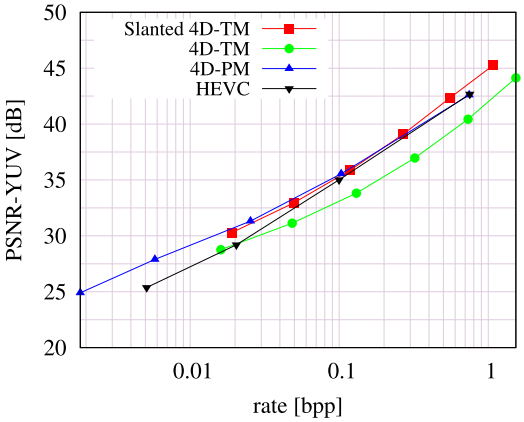
After comparing the proposed Slanted 4D-TM codec with its direct competitors, i.e., the already standardized JPEG Pleno Light Field coding modes, this section intends to compare it with state-of-the-art light field codecs at large. In practice, all these additional benchmarks involve coding standards heavily burdened by royalties' licensing, thus bringing another dimension to the comparison beyond compression efficiency. Since JPEG standards are traditionally licensing-free, these benchmarks do not constitute real solutions as JPEG Pleno coding modes, yet are still relevant to fully assess the RD performance landscape. For the *Tarot* and *Laboratory1* light fields, no comparisons could be made for three of the additional benchmarks since no RD performance results are available.

Table 4 shows the Bjøntegaard delta rate [51] for all test light fields using the HEVC benchmark [8] as reference. These results allow deriving the following conclusions:

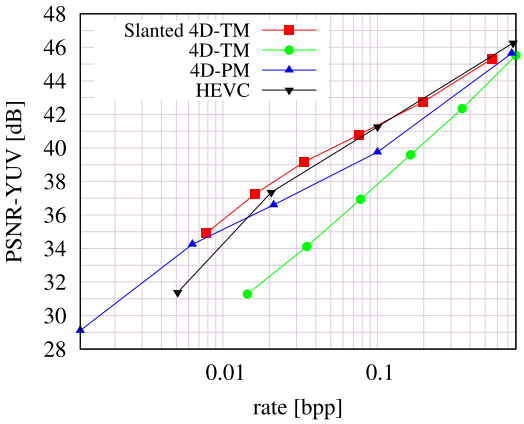
- In terms of RD performance, the proposed Slanted 4D-TM codec outperforms the HEVC benchmark for all the high and medium space-view redundancy light fields. The BD-Rate gains over the HEVC benchmark are significant, with an average gain of 33.34% for the high space-view redundancy light fields and 11.51% for the medium space-view redundancy light fields (see Table 4), showing that both the 4D-TM and Slanted 4D-TM codecs have superior RD performance when coding these types of light fields.
- In terms of RD performance, the proposed Slanted 4D-TM codec outperforms the HEVC codec for the low

TABLE 4. BD-Rate (%) regarding the HEVC benchmark [8].

	Slanted 4D-TM	4D-TM	4D-PM	FDL-hierarchical	MVI-VVC	WaSPR
Bikes	−36.89	−36.70	0.35	−78.55	−65.49	−48.56
Danger	−45.22	−44.56	−16.11	−75.16	−63.63	−53.16
Fountain	−23.44	−22.22	53.20	−76.48	−62.31	−43.35
Pillars	−27.80	−23.00	−13.01	−74.47	−66.23	N/A
Average	−33.34	−31.62	6.11	−76.17	−64.42	−48.36
Sideboard	−13.00	63.29	−19.57	−43.46	−72.48	−45.11
Tarot	−10.01	203.74	49.65	N/A	N/A	N/A
Average	−11.51	133.52	15.04	−43.46	−72.48	−45.11
Greek	−35.41	64.88	−17.47	−40.02	−64.79	−25.37
Laboratory1	102.94	263.59	131.09	N/A	N/A	N/A
Average	33.77	164.24	56.81	−40.02	−64.79	−25.37
Total Average	−7.68	58.63	21.02	−64.69	−65.83	−43.11

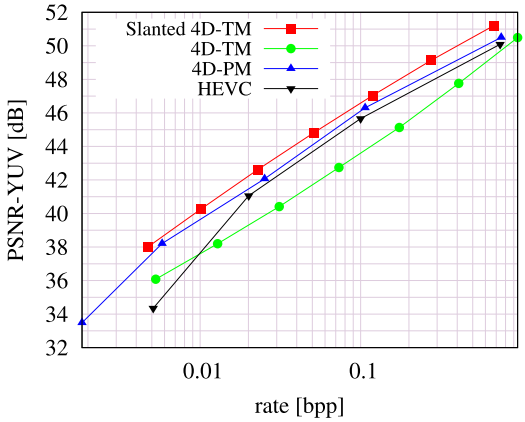


(a) *Sideboard*.

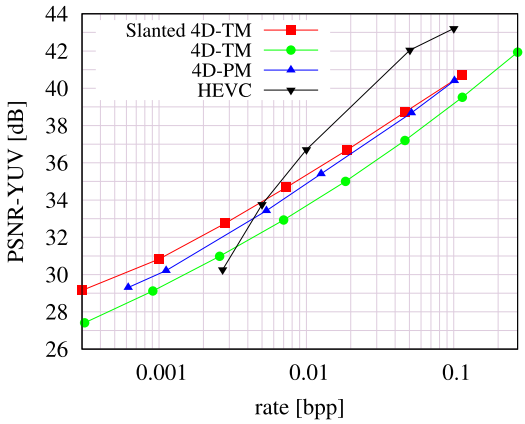


(b) *Tarot*.

FIGURE 16. RD performances for the medium space-view redundancy ***Sideboard*** and ***Tarot*** light fields.



(a) *Greek*.



(b) *Laboratory1*.

FIGURE 17. RD performance for the low space-view redundancy ***Greek*** and ***Laboratory1*** light fields.

space-view redundancy *Greek* light field at all rates. The Slanted 4D-TM BD-Rate gains over the HEVC benchmark are significant, with a BD-Rate gain of 35.41% (Table 4) for the *Greek* light field.

Figs. 17a and 17b, which show the RD performance for the *Greek* and *Laboratory1* light fields, respectively, clearly illustrate that the Slant transform is very efficient in improving the RD performance for the *Greek* light

field when compared with the HEVC RD performance. However, for the *Laboratory1* light field, the proposed codec only provides a better RD performance than HEVC for the low rates, offering a poorer compression efficiency from medium rates onwards. This performance is likely to be improved by using a larger search range for the Slant transform slope σ, as detailed in Subsection IV-E.

- In terms of RD performance, the FDL-hierarchical [36] benchmark, described in Section II, outperforms the HEVC benchmark and the proposed Slanted 4D-TM codec for all test light fields. Its depth-layered representation allied to the sophisticated HEVC [19] coding tools offer large BD-Rates gains.
- In terms of RD performance, the MVI-VVC [24] benchmark, described in Section II, outperforms the HEVC benchmark and the proposed Slanted 4D-TM codec for all test light fields. The proposed MVI structure favors disparity estimation as the target views are close to the reference view, offering large BD-Rate gains (Table 4) due to the use of very efficient standard coding solutions, such as HEVC [19] and VVC [20].
- In terms of RD performance, the WaSPR [45] benchmark, described in Section II, outperforms the HEVC benchmark for all test light fields. It uses HEVC [19] to encode the reference views, the depth map, and the residuals. In this coding method, depth accuracy is of paramount importance as it is used to create estimations of the non-reference views after improved residual coding. It also offers higher BD-Rate gains for the majority of light fields used in this work. However, it is worth noting that, although *Greek* is a computer-generated light field with a very accurate depth map, which favors depth data-dependent light field coding methods such as the WaSPR codec [45], the proposed royalty-free and depth agnostic Slanted 4D-TM codec offers RD performance superior to the one of WaSPR (Table 4).

In summary, the proposed Slanted 4D-DCT codec is already largely outperforming the HEVC benchmark, although still not other benchmarks which are out of consideration as JPEG standards due to the fact that they heavily exploit royalty-burdened standards, such as HEVC and VVC.

VI. CONCLUSION

Light fields are an emerging visual modality able to offer more realistic and immersive experiences in multiple applications. Due to the large amount of data associated with rich light fields, efficient coding is naturally mandatory. Since many of the application domains where light fields may find applications ask for interoperability, light field coding standards are needed. This was the motivation for the development of the JPEG Pleno Light Coding standard which has already specified two coding modes, notably the 4D-TM and 4D-PM coding modes, which are effective for light fields in different baseline ranges. In this context, this paper proposes a new coding mode, extending the 4D-TM coding mode, so-called Slanted 4D-TM, that is able to effectively outperform, in terms of RD performance, both the available JPEG Pleno Light Field coding modes, for all baselines. This improved RD performance behavior justifies specifying an additional JPEG Pleno Light Field coding mode which would provide benefits not only in terms of added compression but also in reduced complexity, e.g., memory footprint, since a single coding mode could be implemented and no coding mode selection would have to be made at coding time depending on the light field baseline. The new coding mode is based on the idea of geometrically processing the light field 4D blocks using a Slant transform that is able to create blocks that may be more efficiently energy-compacted by the separable and orthogonal 4D-DCT already included in the coding pipeline; moreover, the new coding mode does not ask for depth maps. Finally, the novel Slanted 4D-TM mode is likely royalty-free as wished for JPEG standards and different from alternative solutions in the literature.

REFERENCES

[1] E. H. Adelson and J. R. Bergen, "The plenoptic function and the elements of early vision," in *Computational Models of Visual Processing*, M. Landy and J. A. Movshon, Eds. Cambridge, MA, USA: MIT Press, 1991, pp. 3–20.

[2] *JPEG Website*. Accessed: Aug. 31, 2022. [Online]. Available: https://jpeg.org

[3] *Information Technology Plenoptic Image Coding System (JPEG Pleno) Part 2: Light Field Coding*, Standard ISO/IEC 21794-2:2021, Apr. 2021.

[4] T. Ebrahimi, S. Foessel, F. Pereira, and P. Schelkens, "JPEG Pleno: Toward an efficient representation of visual reality," *IEEE MultimediaMag.*, vol. 23, no. 4, pp. 14–20, Oct. 2016.

[5] M. Levoy and P. Hanrahan, "Light field rendering," in *Proc. 23rd Annu. Conf. Comput. Graph. Interact. Techn.*, New York, NY, USA, 1996, pp. 31–42.

[6] R. C. Bolles, H. H. Baker, and D. H. Marimont, "Epipolar-plane image analysis: An approach to determining structure from motion," *Int. J. Comput. Vis.*, vol. 1, no. 1, pp. 7–55, 1987.

[7] D. G. Dansereau, "Plenoptic signal processing for robust vision in field robotics," Ph.D. thesis, Dept. Eng. Inf. Technol., School Aerosp., Mech. Mechtron. Eng., Austral. Centre Field Robot., Univ. Sydney, NSW, Australia, 2014.

[8] F. Pereira, C. Pagliari, E. A. B. da Silva, I. Tabus, H. Amirpour, M. Bernardo, and A. and Pinheiro, *Information Technology—JPEG Pleno Light Field Coding Common Test Conditions V3.3*, Standard ISO/IEC JTC 1/SC29/WG1N84025, 84th JPEG Meeting, Brussels, Belgium, 2019.

[9] C. Pagliari, S. Mahmoudpour, F. Pereira, E. A. B. da Silva, A. Pinheiro, S. Zhao, Y. Peng, and H. Amirpour, *Information Technology—JPEG Pleno Light Field Quality Assessment Common Test Conditions V1.0*, ISO/IEC JTC 1/SC 29/WG1N100335, 97th JPEG Meeting, 2022.

[10] *Information Technology—JPEG Pleno Final Call for Contributions on Subjective Light Field Quality Assessment*, ISO/IEC JTC 1/SC 29/WG1N100307, 97th JPEG Meeting, 2022.

[11] C. Conti, L. D. Soares, and P. Nunes, "Dense light field coding: A survey," *IEEE Access*, vol. 8, pp. 49244–49284, 2020.

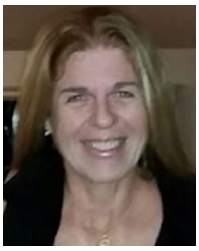
[12] J.-X. Chai, X. Tong, S.-C. Chan, and H.-Y. Shum, "Plenoptic sampling," in *Proc. 27th Annu. Conf. Comput. Graph. Interact. Techn.*, 2000, Art. no. 307318.

[13] *Technical Report of the Joint Ad Hoc Group for Digital Representations of Light/Sound Fields for Immersive Media Applications*, Standard ISO/IEC JTC1/SC29/WG1, 2016.

[14] C. Brites, J. Ascenso, and F. Pereira, "Lenslet light field image coding: Classifying, reviewing and evaluating," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 1, pp. 339–354, Jan. 2021.

[15] S.-H. Tsang, Y.-L. Chan, and W. Kuang, "Standard-compliant HEVC screen content coding for raw light field image coding," in *Proc. 13th Int. Conf. Signal Process. Commun. Syst. (ICSPCS)*, Dec. 2019, pp. 1–6.
[16] S.-H. Tsang, Y.-L. Chan, and W. Kuang, "Mode skipping for HEVC screen content coding via random forest," *IEEE Trans. Multimedia*, vol. 21, no. 10, pp. 2433–2446, Oct. 2019.
[17] S.-H. Tsang, W. Kuang, Y.-L. Chan, and W.-C. Siu, "Reduced-complexity intra block copy (IntraBC) mode with early CU splitting and pruning for HEVC screen content coding," *IEEE Trans. Multimedia*, vol. 21, no. 2, pp. 269–283, Feb. 2019.
[18] *Advanced Video Coding for Generic Audiovisual Services, Record ITU-T H.264 and ISO/IEC 14496-10 Information Technology—Coding of Audio-Visual Objects—Part 10: Advanced Video Coding (MPEG-4 AVC)*, Standard ISO/IEC 14496-10, 2014.
[19] *High Efficiency Video Coding, Rec. ITU-T H.265*, Standard ISO/IEC 23008-2, ITU-T & ISO/IEC, 2013.
[20] *Working Draft 4 of Versatile Video Coding*, Standard ISO/IEC JTC 1/SC29/WG11 N18274, JVET of ITU-T SG 16 WP3, 13th Meeting, Marrakech, Morocco, Jan 2019.
[21] H. Amirpour, M. Pereira, and A. Pinheiro, "High efficient snake order pseudo-sequence based light field image compression," in *Proc. Data Compress. Conf.*, Mar. 2018, p. 397.
[22] F. Dai, J. Zhang, Y. Ma, and Y. Zhang, "Lenselet image compression scheme based on subaperture images streaming," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2015, pp. 4733–4737.
[23] R. J. S. Monteiro, N. M. M. Rodrigues, S. M. M. Faria, and P. J. L. Nunes, "Light field image coding based on hybrid data representation," *IEEE Access*, vol. 8, pp. 115728–115744, 2020.
[24] H. Amirpour, A. Pinheiro, M. Pereira, F. J. P. Lopes, and M. Ghanbari, "Efficient light field image compression with enhanced random access," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 18, no. 2, pp. 1–18, Mar. 2022.
[25] A. Vetro, T. Wiegand, and G. J. Sullivan, "Overview of the stereo and multiview video coding extensions of the H.264/MPEG-4 AVC standard," *Proc. IEEE*, vol. 99, no. 4, pp. 626–642, Apr. 2011.
[26] G. Tech, Y. Chen, K. Müller, J.-R. Ohm, A. Vetro, and Y.-K. Wang, "Overview of the multiview and 3D extensions of high efficiency video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 35–49, Jan. 2016.
[27] W. Ahmad, M. Ghafoor, S. A. Tariq, A. Hassan, M. Sjostrom, and R. Olsson, "Computationally efficient light field image compression using a multiview HEVC framework," *IEEE Access*, vol. 7, pp. 143002–143014, 2019.
[28] W. Tu, X. Jin, L. Li, C. Yan, Y. Sun, M. Xiao, W. Han, and J. Zhang, "Efficient content adaptive plenoptic video coding," *IEEE Access*, vol. 8, pp. 5797–5804, 2020.
[29] H. Han, J. Xin, and Q. Dai, "Plenoptic image compression via simplified subaperture projection," in *Advances in Multimedia Information Processing—PCM 2018*. Cham, Switzerland: Springer, Sep. 2018, pp. 274–284.
[30] W. Ahmad, S. Vagharshakyan, M. Sjostrom, A. Gotchev, R. Bregovic, and R. Olsson, "Shearlet transform-based light field compression under low bitrates," *IEEE Trans. Image Process.*, vol. 29, pp. 4269–4280, 2020.
[31] Y. Zhang, W. Dai, Y. Li, C. Li, J. Hou, J. Zou, and H. Xiong, "Light field compression with graph learning and dictionary-guided sparse coding," *IEEE Trans. Multimedia*, early access, Feb. 28, 2022, doi: 10.1109/TMM.2022.3154928.
[32] B. Heriard-Dubreuil, I. Viola, and T. Ebrahimi, "Light field compression using translation-assisted view estimation," in *Proc. Picture Coding Symp. (PCS)*, Nov. 2019, pp. 1–5.
[33] Y.-H. Chao, H. Hong, G. Cheung, and A. Ortega, "Pre-demosaic graph-based light field image compression," *IEEE Trans. Image Process.*, vol. 31, pp. 1816–1829, 2022.
[34] M. Le Pendu, C. Guillemot, and A. Smolic, "A Fourier disparity layer representation for light fields," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5740–5753, Nov. 2019.
[35] E. Dib, M. L. Pendu, and C. Guillemot, "Light field compression using Fourier disparity layers," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 3751–3755.
[36] M. L. Pendu, C. Ozcinar, and A. Smolic, "Hierarchical Fourier disparity layer transmission for light field streaming," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2020, pp. 2606–2610.
[37] T. Zhong, X. Jin, L. Li, and Q. Dai, "Light field image compression using depth-based CNN in intra prediction," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 8564–8567.
[38] M. Singh and R. M. Rameshan, "Learning-based practical light field image compression using a disparity-aware model," in *Proc. Picture Coding Symp. (PCS)*, Bristol, U.K., Jun. 2021, pp. 1–5.
[39] *Information Technology—JPEG AI Common Training and Test Conditions*, WG1, 94th JPEG Meeting, ISO/IEC JTC 1/SC29/WG1N100106, 2022.
[40] *Information Technology—Corrections and Additions to the JPEG AI Common Training and Test Conditions*, WG1, 95th JPEG Meeting, Standard ISO/IEC JTC 1/SC29/WG1N100106, 2022.
[41] L. Cruz, *Information Technology—Corrections and Additions to the JPEG AI Common Training and Test Conditions*, Standard ISO/IEC JTC 1/SC29/WG1N100276, 95th JPEG Meeting, 2022.
[42] G. De Oliveira Alves, M. B. De Carvalho, C. L. Pagliari, P. G. Freitas, I. Seidel, M. P. Pereira, C. F. S. Vieira, V. Testoni, F. Pereira, and E. A. B. Da Silva, "The JPEG Pleno light field coding standard 4D-form mode: How to design an efficient 4D-native codec," *IEEE Access*, vol. 8, pp. 170807–170829, 2020.
[43] P. Astola and I. Tabus, "WaSP: Hierarchical warping, merging, and sparse prediction for light field image compression," in *Proc. 7th Eur. Workshop Vis. Inf. Process. (EUVIP)*, Nov. 2018, pp. 1–6.
[44] D. Taubman and M. Marcellin, *JPEG2000 Image Compression Fundamentals, Standards and Practice*. Cham, Switzerland: Springer, 2013.
[45] P. Astola and I. Tabus, "Coding of light fields using disparity-based sparse prediction," *IEEE Access*, vol. 7, pp. 176820–176837, 2019.
[46] *Information Technology Plenoptic Image Coding System (JPEG Pleno) Part 2: Light Field Coding Amendment 1: Profiles and Levels for JPEG Pleno Light Field Coding System*, Standard ISO/IEC 21794-2:2021/AMD 1:2021, Aug. 2021.
[47] C. Perra, P. Astola, E. A. B. Da Silva, H. Khanmohammad, C. Pagliari, P. Schelkens, and I. Tabus, "Performance analysis of JPEG Pleno light field coding," in *Proc. SPIE*, vol. 11137, pp. 402–413, Sep. 2019.
[48] T. Leistner, H. Schilling, R. Mackowiak, S. Gumhold, and C. Rother, "Learning to think outside the box: Wide-baseline light field depth estimation with EPI-shift," in *Proc. Int. Conf. 3D Vis. (3DV)*, Sep. 2019, pp. 1–12.
[49] M. P. Pereira, G. Alves, C. L. Pagliari, M. B. de Carvalho, E. A. B. da Silva, and F. Pereira, "A geometric space-view redundancy descriptor for light fields: Predicting the compression potential of the JPEG Pleno light field datasets," in *Proc. IEEE 19th Int. Workshop Multimedia Signal Process. (MMSP)*, Oct. 2017, pp. 1–6.
[50] *JPEG Pleno Light Field Datasets According to Common Test Conditions*. Accessed: Aug. 31, 2022. [Online]. Available: https://jpeg.org/plenodb/lf/plenolf
[51] G. Bjøntegaard, *Calculation of Average PSNR Differences Between RD-Curves*, document VCEG-M33, ITU-T, 2001.
[52] *Anserw*. Accessed: Aug. 31, 2022. [Online]. Available: https://github.com/Anserw/Bjontegaardmetric
[53] *Slanted 4D-TM*. Accessed: Oct. 1, 2023. [Online]. Available: https://github.com/murilotet/muleslant
[54] *JPEG Pleno Light Field Verification Model 2.1*. Accessed: Aug. 31, 2022. [Online]. Available: https://gitlab.com/wg1/jpeg-pleno-vm
[55] *x265 Website*. Accessed: Aug. 31, 2022. [Online]. Available: https://www.x265.org/

MURILO BRESCIANI DE CARVALHO (Member, IEEE) was born in Petrópolis, Brazil, in 1964. He received the B.E. degree in electrical engineering from Universidade Federal Fluminense (UFF), Brazil, in 1986, the M.Sc. degree in electrical engineering (telecommunications systems) from Pontifícia Universidade Católica do Rio de Janeiro (PUC-RJ), in 1994, and the D.Sc. degree in electrical engineering (signal processing) from Universidade Federal do Rio de Janeiro (COPPE/UFRJ), in 2001. Since 1994, he has been with the Department of Telecommunications Engineering, UFF. His research interests include digital image/video processing, source/channel coding, and digital signal processing.



CARLA L. PAGLIARI (Senior Member, IEEE) received the Ph.D. degree in electronic systems engineering from the University of Essex, U.K., in 2000. In 1983 and 1985, she was with TV Globo, Rio de Janeiro, Brazil. From 1986 to 1992, she was a Researcher with Instituto de Pesquisa e Desenvolvimento, Rio de Janeiro. Since 1993, she has been with the Department of Electrical Engineering, Military Institute of Engineering (IME), Rio de Janeiro. Her research interests include light field signal processing and coding, image and video processing, and computer vision. She is a member of the ISO/IEC JTC1/SC29/WG1 (JPEG) Standardization Committee and the Chair of the Ad-Hoc Group on JPEG Pleno Light Field.

GUSTAVO DE O. E. ALVES (Member, IEEE) was born in Cachoeiro de Itapemirim, Brazil. He received the degree in electronics engineering from Universidade Federal do Rio de Janeiro (UFRJ), Brazil, in 2014, and the M.Sc. degree in electrical engineering from Universidade Federal do Rio de Janeiro (COPPE/UFRJ), in 2019, where he is currently pursuing the D.Sc. degree in electrical engineering. From 2011 to 2018, he was a TV Systems Researcher with TV Globo, Rio de Janeiro, Brazil. Since 2019, he has been a Digital TV Engineer with Globoplay, Rio de Janeiro. His research interests include digital television and digital signal/image/video processing.

COLAS SCHRETTER (Member, IEEE) received the M.Sc. degree (Hons.) in computer science, in 2004, and the master's degree in advanced studies from Université libre de Bruxelles, Belgium, in 2006. He won a Marie Curie ERG grant with RWTH Aachen University and was a Visiting Researcher with Politecnico di Milano, Italy. He was selected for the M+Vision 2012 Fellowship with MIT, Cambridge, USA. Since 2013, he has been a Senior Research Scientist with the ETRO Department, Vrije Universiteit Brussel. His research interests include problems in high-dimensional image formation from a large corpus of measurements using physically based models, regularized inverse methods, and advanced statistical estimation techniques. He received the Marie Curie EST fellowship for his Ph.D. thesis, awarded magna cum laude, in 2010.

PETER SCHELKENS (Member, IEEE) holds a professorship with the Department of Electronics and Informatics, Vrije Universiteit Brussel, Belgium. He is currently a Principal Investigator with imec, Belgium. In 2013, he received an EU ERC Consolidator grant focusing on digital holography. He is the co-editor of the books *The JPEG 2000 Suite* (Wiley, 2009) and *Optical and Digital Image Processing* (Wiley, 2011). He is the Chair of the Plenoptic Coding and Quality Subgroup of the ISO/IEC JTC1/SC29/WG1 (JPEG) Standardization Committee.

FERNANDO PEREIRA (Fellow, IEEE) has been the MPEG Requirements Subgroup Chair. He is currently with the Department of Electrical and Computer Engineering, Instituto Superior Técnico and Instituto de Telecomunicações, Lisbon, Portugal. He is also the JPEG Requirements Subgroup Chair. Recently, he has been one of the key designers of the JPEG Pleno and JPEG AI standardization projects. He has contributed more than 300 papers in international journals, conferences, and workshops, and made several tens of invited talks and tutorials at conferences and workshops. His research interests include video analysis, representation, coding, description and adaptation, and advanced multimedia services. He was an IEEE Distinguished Lecturer, in 2005, and elected as a fellow of IEEE in 2008 for "Contributions to object-based digital video representation technologies and standards." Since 2013, he has been a EURASIP Fellow for "Contributions to digital video representation technologies and standards." Since 2015, he has been a fellow of IET. He is or has been a member of the Editorial Board of the *Signal Processing Magazine*, an Associate Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON MULTIMEDIA, and *IEEE Signal Processing Magazine*. He has been elected to serve on the Signal Processing Society Board of Governors in the Capacity of Member-at-Large (2012) and (2014–2016). He has been the Vice President of the IEEE Signal Processing Society, from 2018 to 2020. He has also been elected to serve on the European Signal Processing Society Board of Directors (2015–2018). He is an Area Editor of the *Signal Processing: Image Communication* journal and an Associate Editor of the *EURASIP Journal on Image and Video Processing*. From 2013 to 2015, he was the Editor-in-Chief of the IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING.

EDUARDO A. B. DA SILVA (Senior Member, IEEE) was born in Rio de Janeiro, Brazil. He received the degree in electronics engineering from Instituto Militar de Engenharia (IME), Brazil, in 1984, the M.Sc. degree in electrical engineering from Universidade Federal do Rio de Janeiro (COPPE/UFRJ), in 1990, and the Ph.D. degree in electronics from the University of Essex, England, in 1995. He has been a Professor with Universidade Federal do Rio de Janeiro, since 1989. He is the coauthor of four books, including the book *Digital Signal Processing—System Analysis and Design* (Cambridge University Press), with editions in English, Chinese, and Portuguese. His research interests include the fields of signal and image processing, signal compression, 3D videos, computer vision, light fields, and machine learning, together with their applications to telecommunications and the oil and gas industry. He is currently the requirements Vice Chair of the Joint Photographic Experts Group from ISO/IEC. He was an Editor of the Standard ISO/IEC CD 21794-2, JPEG Pleno Plenoptic Image Coding System.