ISCTE IUL
Instituto Universitário de Lisboa

Department of Information Science and Technology

Scalable Light Field Representation and Coding

Ricardo Jorge Santos Monteiro

Thesis specially presented for the fulfillment of the degree of

Doctor in Information Science and Technology

Supervisor:

Dr. Paulo Jorge Lourenço Nunes, Assistant Professor

ISCTE-IUL

Co-supervisor:

Dr. Nuno Miguel Morais Rodrigues, Assistant Professor

ESTG – Polytechnic of Leiria

December 2019

ISCTE IUL
Instituto Universitário de Lisboa

Department of Information Science and Technology

Scalable Light Field Representation and Coding

Ricardo Jorge Santos Monteiro

Thesis specially presented for the fulfillment of the degree of

Doctor in Information Science and Technology

Jury:

President (by delegation from rector of ISCTE-IUL): Dr. Adolfo Cartaxo, Full Professor, ISCTE-IUL

Members of panel:

Dr. Mårten Sjöström, Full Professor, Mid Sweden University

Dr. Fernando Pereira, Full Professor, IST - University of Lisbon

Dr. Pedro Assunção, Coordinator Professor, ESTG - Polytechnic of Leiria

Dr. Luis Cruz, Assistant Professor, FCTUC - University of Coimbra

Dr. Luis Soares, Associate Professor, ISCTE-IUL

Dr. Paulo Nunes, Assistant Professor, ISCTE-IUL

December 2019

Abstract

This Thesis aims to advance the state-of-the-art in light field representation and coding. In this context, proposals to improve functionalities like light field random access and scalability are also presented. As the light field representation constrains the coding approach to be used, several light field coding techniques to exploit the inherent characteristics of the most popular types of light field representations are proposed and studied, which are normally based on micro-images or sub-aperture-images.

To encode micro-images, two solutions are proposed, aiming to exploit the redundancy between neighboring micro-images using a high order prediction model, where the model parameters are either explicitly transmitted or inferred at the decoder, respectively. In both cases, the proposed solutions are able to outperform low order prediction solutions.

To encode sub-aperture-images, an HEVC-based solution that exploits their inherent intra and inter redundancies is proposed. In this case, the light field image is encoded as a pseudo video sequence, where the scanning order is signaled, allowing the encoder and decoder to optimize the reference picture lists to improve coding efficiency.

A novel hybrid light field representation coding approach is also proposed, by exploiting the combined use of both micro-image and sub-aperture-image representation types, instead of using each representation individually.

In order to aid the fast deployment of the light field technology, this Thesis also proposes scalable coding and representation approaches that enable adequate compatibility with legacy displays (e.g., 2D, stereoscopic or multiview) and with future light field displays, while maintaining high coding efficiency. Additionally, viewpoint random access, allowing to improve the light field navigation and to reduce the decoding delay, is also enabled with a flexible trade-off between coding efficiency and viewpoint random access.

Keywords: Light field, Light field coding, Light field representation, Viewpoint scalability, Viewpoint random access

Resumo

Esta Tese tem como objetivo avançar o estado da arte em representação e codificação de campos de luz. Neste contexto, são também apresentadas propostas para melhorar funcionalidades como o acesso aleatório ao campo de luz e a escalabilidade. Como a representação do campo de luz limita a abordagem de codificação a ser utilizada, são propostas e estudadas várias técnicas de codificação de campos de luz para explorar as características inerentes aos seus tipos mais populares de representação, que são normalmente baseadas em micro-imagens ou imagens de sub-abertura.

Para codificar as micro-imagens, são propostas duas soluções, visando explorar a redundância entre micro-imagens vizinhas utilizando um modelo de predição de alta ordem, onde os parâmetros do modelo são explicitamente transmitidos ou inferidos no decodificador, respetivamente. Em ambos os casos, as soluções propostas são capazes de superar as soluções de predição de baixa ordem.

Para codificar imagens de sub-abertura, é proposta uma solução baseada em HEVC que explora a inerente redundância intra e inter deste tipo de imagens. Neste caso, a imagem do campo de luz é codificada como uma pseudo-sequência de vídeo, onde a ordem de varrimento é sinalizada, permitindo ao codificador e decodificador otimizar as listas de imagens de referência para melhorar a eficiência da codificação.

Também é proposta uma nova abordagem de codificação baseada na representação híbrida do campo de luz, explorando o uso combinado dos tipos de representação de micro-imagem e sub-imagem, em vez de usar cada representação individualmente.

A fim de facilitar a rápida implantação da tecnologia de campo de luz, esta Tese também propõe abordagens escaláveis de codificação e representação que permitem uma compatibilidade

adequada com monitores tradicionais (e.g., 2D, estereoscópicos ou multivista) e com futuros monitores de campo de luz, mantendo ao mesmo tempo uma alta eficiência de codificação. Além disso, o acesso aleatório de pontos de vista, permitindo melhorar a navegação no campo de luz e reduzir o atraso na descodificação, também é permitido com um equilíbrio flexível entre eficiência de codificação e acesso aleatório de pontos de vista.

Palavras-chave: Campos de luz, Codificação de campos de luz, Representação de campos de luz, Escalabilidade de ponto de vista, Acesso aleatório de ponto de vista

Acknowledgements

First and foremost, I would like to thank my supervisors Prof. Nuno Rodrigues, Prof. Paulo Nunes and Prof. Sérgio Faria for their guidance and trust throughout the PhD. I'm truly grateful to have such great examples of leadership, altruism and experience by my side along one of the hardest challenges that I ever endured. I hope I'll be able to emulate their values in the future for the many challenges that await me.

I am grateful to my current and former colleagues from *Instituto de Telecomunicações* from both Lisbon and Leiria. A special thanks to Prof. Paulo Correia and Prof. Luís Ducla Soares for being so welcoming and for the very fruitful insights about our research topics and Prof. Fernando Pereira specially for the presentation skills. I would also like express my gratitude to my colleagues and friends Tanmay Verlekar, Francisco Cunha, André Guarda, Pedro Pereira, João Santos, João Carreira, José Filipe, Rui Lourenço and Lucas Thomaz, which I both love and hate at times.

I would like to acknowledge *Instituto de Telecomunicações* for the great working conditions and *Fundação para a Ciência e Tecnologia* for funding this PhD (under the UID/EEA/50008/2013 and UID/EEA/50008/2019 projects and SFRH/BD/136953/2018 grant)

I would like to thank my friends for always being able to put up with me, specially to André Sousa my brother from another mother, Nuno Santos, Nuno Miguel, Vera Pereira and Mariana Borges.

I am deeply grateful to my parents and my sister which always see the best in me, even when I fail to do so. Their support and love is beyond anything I could ever hope for.

Last but not least, I would like to express my gratitude to Iris, for her love and emotional support. May our unconditional love for Nutella continue to unite us (and make us fat) in the future.

Table of Contents

List of Figures

List of Tables

List of Acronyms

AGSP	accurate gradient selective prediction
AMP	asymmetric motion prediction
AMVP	advanced motion vector prediction
AR/VR	augmented and virtual reality
BD	Bjøntegaard delta
BEPC	Base evaluation processing chain
CABAC	context adaptive binary arithmetic coding
CB	coding block
CNN	convolutional neural network
CTB	coding tree block
CTU	coding tree unit
CU	coding unit
DBF	Deblocking filter
DCT	discrete cosine transform
DIBR	depth image-based rendering
DoF	degrees of freedom
DST	discrete sine transform
FOC	focused

FTV	Free-viewpoint television
GAN	generative adversarial network
GAP	gradient adjusted predictor
GT	geometric transformation
HDCA	high-density camera array
HDR	high-dynamic-range
HLRA	homography-based low-rank approximation
HMD	head mounted display
HOP	High order prediction
HR	hybrid LF data representation
HRPL	hybrid reference picture list
LF	Light field
LL	“Raw” Lenslet
LLE	Locally linear embedding
LOP	low order prediction
LSP	least squares prediction
MED	median edge detector
MI	micro-image
MLA	microlens array
MuLE	Multidimensional LF encoder
NCC	normalized cross correlation
PB	prediction block
POC	picture order count
PU	prediction unit

PVS	pseudo video sequence
QP	quantization parameter
QoE	quality of experience
RAP	random access penalty
RD	rate-distortion
RoI	region of interest
RPL	reference picture list
RPS	reference picture selection
SAD	sum of absolute differences
SAI	sub-aperture image
SAO	Sample adaptive offset
SKV	structural key views
SLIC	super linear iterative clustering
SS	self-similarity
SVD	singular value decomposition
2D-DWT	two-dimensional discrete wavelet transform
3D-DWT	three-dimensional discrete wavelet transform
TU	transform unit
UHD	ultra-high definition
UNF	unfocused
VOD	video on demand
WaSP	hierarchical warping, merging and sparse prediction

Chapter 1. Introduction

1.1. Context and motivation

In the last ten years we have witnessed exponential improvements in multimedia enabling technologies, which span from the vastly improved transmission capabilities to the exponential increase in processing power. One of the areas that was influenced the most by these improvements is video content distribution, as shown Figure 1.1. In 2017 video traffic, which includes internet video and IP video on demand (VOD), accounted for 75% of the total global IP traffic and it is expected to increase to 82% by 2022 [1]. Moreover, the preferred device in 2017 was the PC with 41% of global IP traffic, however, the current trend has already shown that smartphones are the dominant device and it is expected that in 2022 they are going to account for 44% of the global IP traffic [1].

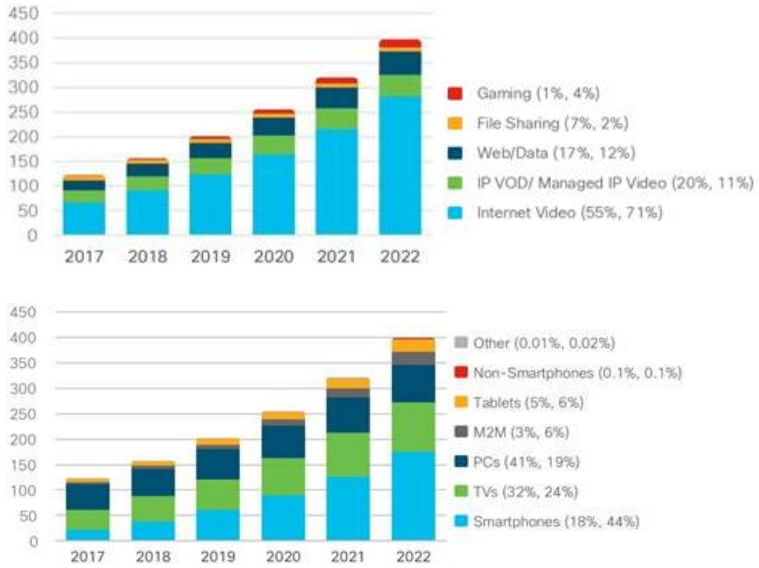


Figure 1.1. Global IP traffic by application (top) and device (bottom) category in Exabytes per month [1].

As mentioned above, the transmission capabilities have improved exponentially, especially when it comes to internet speeds not only for wired connections but also for wireless connections. Technologies like 5G (ITU-T IMT-2020) [2] and Wi-Fi 6 (IEEE 802.11ax) [3] allow for speeds of up to 10 Gbps and 1ms latency over a wireless connection for the end-user, which further potentiate the use of portable devices.

Besides the improvements in display technology, several video characteristics have contributed to the increased user visual quality of experience (QoE) and that also influence the increased volume of video related traffic, notably:

- **Spatial resolution**: In 2022 it is expected that 22% of the global IP video traffic will be 4K ultra-high definition (UHD) [1];
- **Temporal resolution**: Increased refresh rates (over 120Hz) in several type of devices, e.g., computer monitors, TVs, smartphones;
- **Color gamut**: Increased range in terms of color and luminosity components, e.g., high-dynamic-range (HDR);

Several efforts have been made throughout the years to also increase the angular resolution, through technologies such as stereoscopic 3D and autostereoscopic 3D. Although this technology allows for a decent depth perception, the inconvenience of requiring a special type of eyewear to consume 3D content, specifically in the case of stereoscopic 3D, as well as restricting the end-user to specific viewing perspectives reduced the demand for this type of technology. Nowadays alternative media acquisition and display systems have been rapidly progressing towards more immersive media production and consumption.

Light field (LF) is one of the examples of these new immersive media formats. LF is an imaging technology that allows to jointly capture the scene radiance and angular information using single-tier lenslet LF cameras, i.e., with narrow baseline, or by using, for example, a high-density camera array (HDCA), i.e., with a wider baseline. A lenslet LF camera is composed by the standard main lens and sensor, common to 2D cameras, with the addition of a third element: the microlens array (MLA) [4]. The MLA allows the LF camera to capture both spatial and angular information about the light that converges to the sensor [5]. Depending on the LF capturing device, different degrees of freedom are available in terms of both spatial and angular

resolution [6]. Nonetheless, the captured LF information has the ability to convey 3D information about the scene, instead of representing just a single 2D perspective.

The LF technology has recently attracted the interest of many research groups as well as standardization bodies, such as JPEG Pleno [7] and MPEG-I [8], targeting to improve compression efficiency and to normalize LF data representation, as well as other types of new immersive media formats like point cloud, holographic and 360-video.

Technologies that benefit from LF include augmented and virtual reality (AR/VR) and 3D displays that allow realistic depth perception, including continuous motion parallax through the viewing zone. In addition, new post-production functionalities are possible, such as, *a posteriori* refocusing of a scene or changing the scene's viewing perspective.

The large amount of data required to adequately represent a LF scene, when compared to the case of typical 2D pictures, calls for efficient techniques for both transmission and storage of this type of content. To this end, it is mandatory that new coding techniques are developed specifically designed with LF content in mind.

The following sections describe this Thesis objectives and contributions as well as its outline.

1.2. Thesis objectives and contributions

This Thesis aims to advance the state-of-the-art in terms of LF image coding, representation and associated functionalities. In this sense, three major research objectives were defined:

1. **Proposal of LF enhanced coded representations:**

   The representation format of LF has to be carefully defined because it directly influences other processing blocks of the transmission pipeline. LF representation heavily affects how the coding efficiency as well as the types of functionalities that are possible for the end-user. The most popular LF data representations are the Lenslet "Raw" and the 4D LF representations, including their micro-image (MI) and sub-aperture image (SAI) based variants. In order to determine how much does the representation affect LF coding, the following contribution was published [9], where these LF data representations were compared using the HEVC standard:

- R. J. S. Monteiro, N. M. M. Rodrigues, S. M. M. Faria, and P. J. L. Nunes, "Light field image coding: objective performance assessment of Lenslet and 4D LF data representations," in Proc. SPIE, 2018, vol. 10752, pp. 109-125.

This study allowed the authors to assess how the representation affects the coding efficiency results when a state-of-the-art standard video encoder is used without any modifications. Although this solution is not specifically suited for LF coding, it is widely available.

It was concluded that, when LF content is encoded using a SAI-based representation the coding efficiency is higher than when using a MI-based representation. This result makes sense, considering that most of the coding technology is based on exploiting intra and inter image prediction. However, it was observed that when the MI-based representation was being used, the redundancy between the MIs was not exploited by any technique, which contributes to the lack of coding efficiency. Therefore, the authors proposed the following contribution where the proposed LF image codec is able to select the most efficient LF representation for each coding block, either based on MIs or SAIs:

- R. J. S. Monteiro, P. J. L. Nunes, N. M. M. Rodrigues and S. M. M. Faria, "Light Field Image Coding Based on Hybrid Data Representation," IEEE Access (submitted in December 2019).

When each type of representation is selected, the prediction techniques change in order to either exploit the intra and inter SAI redundancy or the intra and inter MI redundancy. This coding solution shows that the combination of both types of representation allows for better coding efficiency than when relying on only one solution individually.

2. Proposal of efficient LF coding algorithms:

Although the combination of both representations leads to better coding results, the authors also proposed LF coding algorithms which are based on only one representation type, i.e., MI or SAI-based representations. One of the first type of solutions that showed high efficiency coding results for MI-based representation LF images was the search algorithm-based solutions. Therefore, a study was conducted on using a combination of two coding approaches to exploit the inter-MI redundancy, namely, self-similarity and locally linear embedding [10]:

- R. J. S. Monteiro, L. F. R. Lucas, C. Conti, P. J. L. Nunes, N. M. M. Rodrigues, S. M. M. Faria, C. L. P. Pagliari, E. A. B. Silva, L. D. Soares, "Light Field HEVC-Based Image Coding using Locally Linear Embedding and Self-Similarity Compensated Prediction," IEEE International Conference on Multimedia and Expo (ICME), 2016, pp. 1-4

Although this study produced a very efficient solution to encode LF images, especially when LF was captured using a focused LF camera, the main denominator of most search-based coding techniques was that these techniques were based on low order prediction, only limited to two degrees of freedom. If complex transformations, such as perspective change between the several MIs, require to be accurately described, prediction methods based on high order prediction were necessary. Consequently, the authors proposed a new block prediction method specifically designed for LF content, which allows geometric transformations between several blocks to be described with up to eight degrees of freedom [11]:

- R. J. S. Monteiro, P. J. L. Nunes, N. M. M. Rodrigues, S. M. M. Faria, "Light Field Image Coding Using High-Order Intra block Prediction," IEEE Journal on Selected Topics in Signal Processing, 2017, Vol. 11, No. 7, pp. 1120-1131.

One of the short comings of the high order prediction-based techniques is the fact that they require more information, i.e. transformation parameters to be transmitted to the decoder, when compared to low order prediction-based techniques. Therefore, the following contribution was published, which is based on performing a training step for several training directions in the encoder side that allows the transformation parameters to be inferred in the decoder, consequently, vastly reducing the amount of information transmitted by the encoder [12]:

- R. J. S. Monteiro, P. J. L. Nunes, S. M. M. Faria, N. M. M. Rodrigues, "Light Field Image Coding using High Order Prediction Training," European Signal Processing Conference (EUSIPCO), 2018, pp. 1845-1849.

Alternatively to the MI-based representation, the SAI-based representation allows for existent video or multiview coding solutions to be used to very efficiently encode the LF image, despite requiring additional processing steps. As mentioned before, the SAI-based

representations are more advantageous when the HEVC standard is straightforwardly applied. Several scanning orders are available in the literature that allowed the conversion of SAIs to a pseudo video sequence, i.e. a sequence of SAIs. However, most conversions are blindly applied as a pre-processing step and the encoder is not aware of the original spatial position of each SAI. The following contribution [13] allows the HEVC to transmit the scanning order applied in pre-processing step, allowing the reference picture selection to be optimized according to the distance between the original spatial positions of each SAI:

- R. J. S. Monteiro, P. J. L. Nunes, S. M. M. Faria, N. M. M. Rodrigues, "Optimized Reference Picture Selection for Light Field Image Coding", European Signal Processing Conference (EUSIPCO), 2019, pp. 1-5. (best student paper award candidate)

3. Proposal of LF coding algorithms that allow for scalability and random access:

In order to enable LF content to be presented on various types legacy displays, such as 2D, and 3D/Stereo displays, as well as on newer LF displays, with different characteristics in terms of spatial and angular resolutions, an efficient scalable codec is proposed. As spatial resolution scalable approaches are already widely available, part of this Thesis is focused on proposing highly efficient coding solution for LF images that also allows high flexibility in terms of viewpoint scalability. Viewpoint scalability allows for not only support for legacy displays but also features like scalable decoding that can be based on factors like available processing power, storage space or network conditions. Additionally, the proposed solution also provides viewpoint random access functionalities. These functionalities are also very advantageous to the end-user, providing features like improved LF navigation and reducing the decoding delay and the computational complexity. This proposed scalable codec is configurable to allow a fine control and a tradeoff between coding efficiency and viewpoint random access:

- R. J. S. Monteiro, P. J. L. Nunes, N. M. M. Rodrigues, S. M. M. Faria, "Efficient Light Field Image Coding with Viewpoint Scalability and Random Access," IEEE Transactions on Circuits and Systems for Video Technology (submitted December 2019).

1.3. Thesis outline

This Thesis proposes several improvements to the state-of-the-art regarding LF representation, coding and functionalities such as scalability and random access.

After this introductory chapter, the remaining of the Thesis is organized as follows:

- **Chapter 2** briefly describes the LF transmission/storage pipeline, which include, LF acquisition, representation, coding, rendering, display and objective evaluation.
- **Chapter 3** reviews the current most relevant coding standards, namely HEVC video coding standard and JPEG Pleno Part 2: LF Coding. Additionally, a review on the state-of-the-art LF image coding approaches is presented.
- **Chapter 4** includes a study on the different LF representations and their effects on LF coding efficiency.
- **Chapter 5** includes the main contributions in terms of LF image coding, based on a MI-based representation.
- **Chapter 6** presents a proposed LF image coding approach which is based on a hybrid representation, including both MI and SAI representations.
- **Chapter 7** proposes a LF image codec based a SAI representation which is able to achieve high coding efficiency as well as functionalities such as viewpoint scalability and random access.
- **Chapter 8** concludes this Thesis discussing the achievements and some future work.

Chapter 2. Light field imaging

In this chapter, the typical full LF storage/transmission pipeline considered in this Thesis is described, from LF acquisition to visualization. As this Thesis is mostly focused on lenslet LF images, the explanation will follow the lenslet LF image storage/transmission pipeline, which presents more processing blocks than pipelines used for LFs acquired by other means.

Figure 2.1 LF imaging storage/transmission pipeline.

As shown in Figure 2.1, the LF storage/transmission pipeline can be composed by 5 steps:

1. **Acquisition**: LFs may be acquired using different types of devices, which allow for different capabilities in terms of spatial, angular and temporal resolution, as described in Section 2.1.
2. **Representation**: The chosen LF representation, may have a heavy influence in the coding efficiency. To illustrate this, the most used LF representations such as the Lenslet "Raw" and the 4D LF data representation are described in Section 2.2. A further study is presented in Chapter 4, which compares several LF representations in terms of coding efficiency.
3. **Encoding/Decoding**: The main standardization initiatives for LF coding and enabled functionalities are briefly discussed in Section 2.3. Additionally, as coding efficiency is the main focus of this research, Chapter 3 presents a state-of-the-art review on LF coding

solutions while Chapters 5, 6 and 7 detail and discuss the author's contributions in this field.

4. **Rendering:** Prior to visualization, the viewpoints are rendered according to the type of used capturing device. Some examples of LF rendering techniques organized by LF capturing device type are discussed in Section 2.4.
5. **Visualization**: LF content may be visualized in various types of devices, that range from standard 2D displays up to full-fledged 4D LF displays, as described in Section 2.5.

In order to evaluate the efficiency of the proposed LF image codecs in terms of objective quality, two processing chains are described in Section 2.6.

2.1. Light field acquisition

A regular LF image can be defined as a four-dimensional signal, as defined by (2.1):

$$LF(h, v, x, y) \tag{2.1}$$

where two dimensions describe the spatial positions, i.e., (x, y) and the two remaining dimensions describe the angular positions i.e., (h, v) [14]. However, conventional camera sensors can only measure the information from two dimensions, i.e., the two spatial dimensions. Consequently, the remaining information required, i.e., angular information, can only be derived by having multiple 2D perspectives, which may be achieved by capturing a LF image. The most popular approaches to capture LF images may be grouped into two major categories [15]:

1. **Based on conventional cameras (multiple or single):** These approaches include the well-known HDCA, which capture a LF image by using an array of conventional cameras, as well as using a single conventional camera to capture several viewpoints at different perspectives within a specific time interval;
2. **Based on lenslet LF cameras:** This type of camera uses a conventional camera sensor, as well as an MLA to capture both spatial and angular information. There are essentially two types of lenslet LF cameras – unfocused [5] and focused [6], i.e., plenoptic 1.0 and plenoptic 2.0, respectively – that provide different capabilities in terms of spatial and angular resolution.

The next sections describe each category in more detail, as well as listing the advantages and disadvantages of each type of LF capturing device.

2.1.1. Conventional light field acquisition

The most common way to capture LF images using conventional 2D cameras, is to have a multi-sensor solution like an array of cameras capturing a scene. This solution is also used in multiview video applications, however, in such case, typically only one-dimensional horizontal camera array is used. Figure 2.2 shows two examples of 2D camera arrays used for LF acquisition from Stanford University [16] and Fraunhofer [17], respectively.

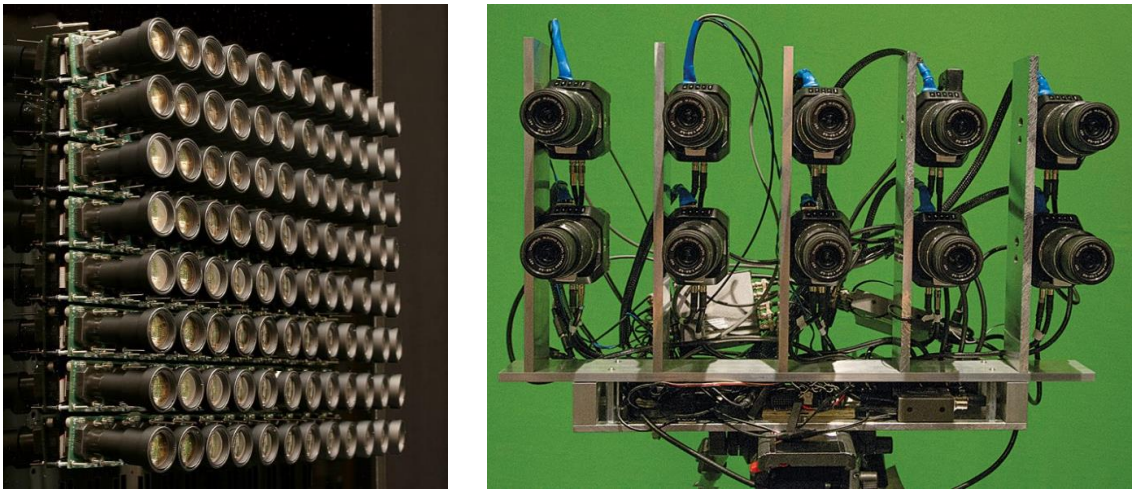


Figure 2.2 Examples of camera arrays from Stanford University [16] (left) and Fraunhofer [17] (right).

The main advantage of using an array of cameras 2D to capture a LF is the fact that it allows the cameras to have different configurations, such as parallel or convergent camera arrangements. Such freedom also allows the baseline between the several cameras to be adjusted according to the required application. Additionally, the spatial and temporal resolution of each viewpoint is only limited by the camera hardware.

Alternatively, a solution involving a time-sequential capture approach using a single image sensor is also possible, in case of static scenes. This approach consists in capturing the desired viewpoints using a conventional 2D camera mounted on a mechanical gantry on different time intervals, as can be seen in Figure 2.3 [18].

Although it is not possible to capture video using this 2D camera gantry, it allows for a potentially more cost-effective approach when compared to a 2D camera array. Additionally,

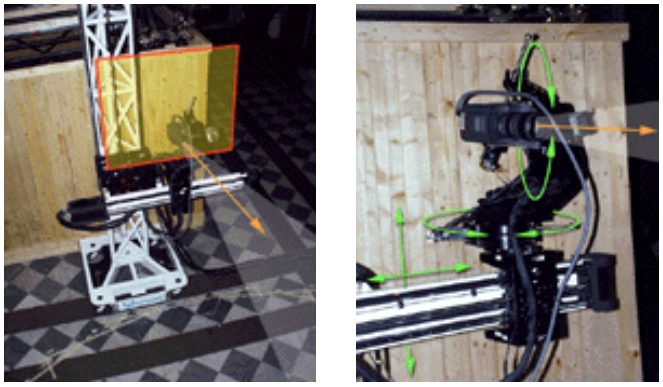


Figure 2.3. 2D camera gantry from Stanford University [18].

the solution used by Stanford University is able to position the camera with four degrees of freedom (DoF), while achieving sub-millimetric accuracy [18].

2.1.2. Lenslet light field acquisition

A lenslet LF camera is a device based on a single image sensor approach that can be categorized as multiplexed imaging [19]. This type of approach works by multiplexing the angular and the spatial domain, effectively imposing a tradeoff to be made between angular and spatial resolution being captured by the camera sensor. This is achieved by using a MLA between the sensor and the main lens. Some example of this type of devices include the hand-held Lytro Illum [20] and the Raytrix R42 [21] LF cameras shown in Figure 2.4.

Figure 2.4. Examples of lenslet LF camers, Lytro Illum [20] (left) and Raytrix R42 [21] (right).

These are representative of the two different models of LF cameras, as the Lytro Illum is an unfocused LF camera and the Raytrix R42 is a focused LF camera. The type of model is based on the position of the camera sensor and the MLA relatively to the main lens, as illustrated in Figure 2.5. With different distances, different samplings of the LF can be performed, which characterizes the lenslet LF camera model [6].

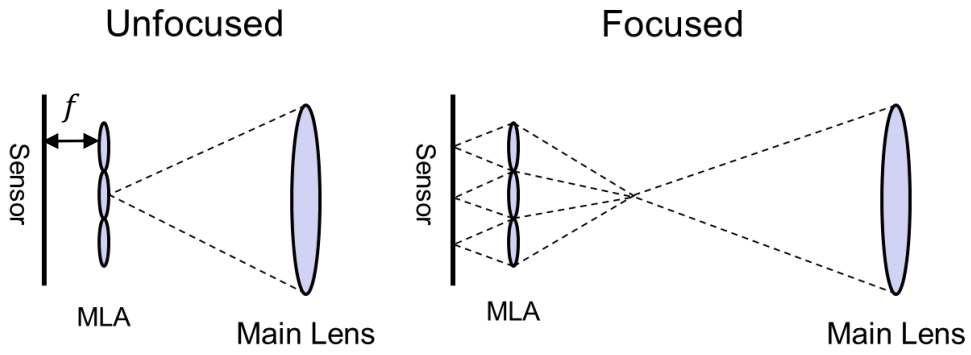


Figure 2.5. Unfocused (left) and focused (right) LF camera models.

In the classic unfocused camera model case (Figure 2.5 left), the sensor is one focal distance away from the MLA. Thus, the MLA is focused at infinity, i.e., the light rays that reach the MLA are parallel [5]. Consequently, the MLA is completely defocused from the main lens image plane. Therefore, each microlens only captures angular information, meaning that each pixel, within the MI, corresponds to a different angle or viewpoint [5]. In the focused lenslet LF camera model (Figure 2.5 right), the sensor is away from the MLA focal distance and the MLA is focused on the main lens image plane, allowing for each microlens to generate a focused MI. This feature allows a higher spatial resolution for rendering, since more than one pixel can be extracted from each MI in the rendering process [6].

2.1.3. Comparison of light field acquisition methods

In order to better understand the advantages and disadvantages of the several LF acquisition methods, Table 2.1 shows a comparison in terms of usability and resolution (spatial and angular).

Although the hand-held LF camera has potentially the highest usability and also a potentially high angular resolution when compared to camera arrays, it has a very low baseline, between neighboring microlenses. This is a limitation when trying to generate a 3D representation of a scene. If a higher baseline is required in order to capture a specific scene, then the only alternative is to use a 2D camera array or a 2D camera gantry. Regardless, some LF representation and coding solutions are independent with relation to the type of LF acquisition method, which simplifies the remainder of the storage/transmission pipeline.

Table 2.1. Advantages and disadvantages between the several LF acquisition methods

LF acquisition methods	Advantages	Disadvantages
2D camera array	Spatial resolution is independent from the angular resolution	Low portability Requires calibration and synchronization before being used Cost scales with the angular resolution
2D camera gantry	Spatial resolution is independent from the angular resolution Mostly unlimited angular resolution Cost is independent from angular resolution Does not allow video capture	Low portability Requires calibration and synchronization before being used
Unfocused LF camera	High portability Mostly "plug and play" High angular resolution	Low spatial resolution
Focused LF camera	High portability Mostly "plug and play High spatial resolution	Low angular resolution

2.2. Light field representation

After acquiring the LF, it is necessary to apply a pre-processing step to mitigate some problems related to the acquisition step, e.g., vignetting., This pre-processing step will define the LF representation, which is also will influence the following steps of the storage/transmission pipeline, such as the LF encoding step. These processing blocks that comprise the LF pre-processing step are shown in Figure 2.6.

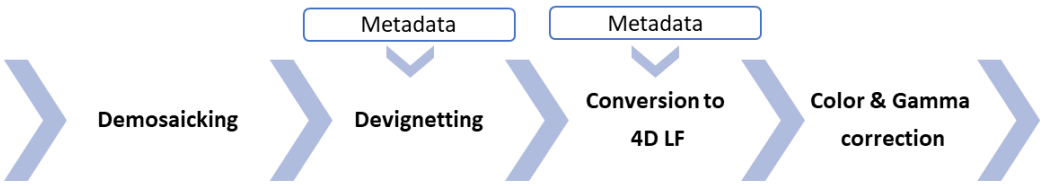


Figure 2.6. Typical processing blocks that compose the LF pre-processing step.

It is worth mentioning that the only step that is mandatory is the demosaicking, however the remaining steps have several advantages that should be considered. The processing blocks present in Figure 2.6 can be described as follows:

- **Demosaicking (mandatory):** The process of reconstructing a full-resolution color image from the sampled data acquired by a digital camera that applies a color filter array to a single sensor. This processing block allows the generation of a RGB image from the raw image, as shown in Figure 2.7.

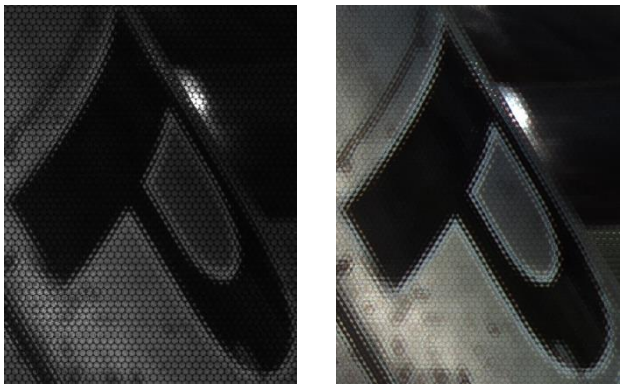


Figure 2.7. LF image before (left) and after (right) the demosaicking step.

- **Devignetting (optional):** It is the process used to eliminate vignetting, very common in optics, which is characterized by darkening of the image corners. In LFs captured using MLA-based LF cameras, this problem is more noticeable because it may occur on every single microlens, as shown in Figure 2.8. Consequently, some rendered views may be darker as well, if this characteristic is not mitigated.

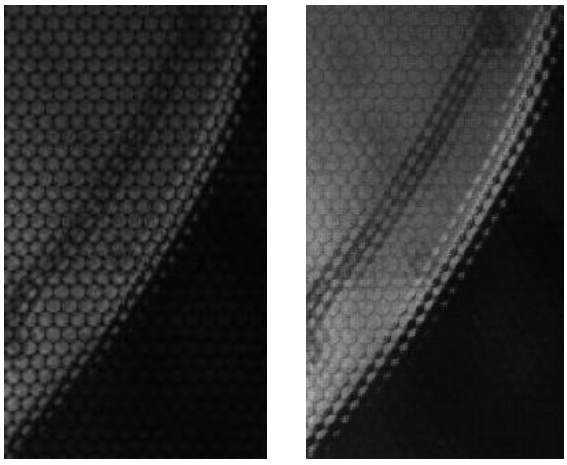


Figure 2.8. LF image before (left) and after (right) the devignetting step.

- **Conversion to 4D LF (optional):** This data conversion corresponds to organizing the LF image as in (2.1). If the LF data was captured with a lenslet LF camera, the conversion to 4D LF requires some metadata information about the camera. Although less compact than the "Raw" Lenslet data representation, the 4D LF data representation facilitates view rendering.
- **Color & Gamma correction (optional)**: This processing block is used to fix color and illumination issues, making the images look more similar to the captured scene, as illustrated in Figure 2.9.

Figure 2.9. Rendered central view before (left) and after (right) color ad gamma correction.

The lenslet LF image is typically represented either as the "Raw" Lenslet data or using the 4D LF format. The following sections define how each LF data representation is generated using the processing blocks shown in Figure 2.6.

2.2.1. Raw lenslet light field representation

The "Raw" Lenslet (LL) data representation, as shown in Figure 2.10, is generated by applying only two pre-processing steps to the captured LF image, that are demosaicking and devignetting. The devignetting process normally uses metadata information about the LF camera used to capture the LF image. In the case of the Lytro Illum for example, the metadata includes white images. As shown in Figure 2.11, these images were acquired using a white diffuser or a white scene [14], being used to estimate the center of each microlens, which is considered the brightest pixel within each microlens (marked in red in Figure 2.11). The center location of each microlens can also be used for the view's rendering process of the LL LF image or to convert it to the 4D LF data representation.

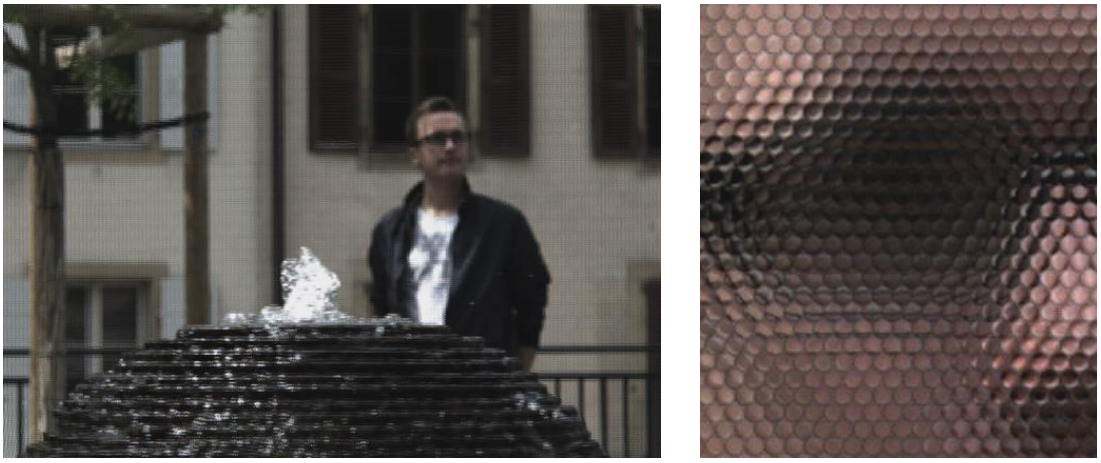


Figure 2.10. Full LF image (left) and zoom of the same LF image (right), using the ”Raw” Lenslet data representation.

Figure 2.11. Microlens center estimation using the Lytro Illum white images [14].

2.2.2. 4D light field representation

The 4D LF data representation organizes the LF image into a four-dimensional array of data that can be defined as $LF(h, v, x, y)$, i.e., as it was in (2.1). It comprises a stack of SAIs that is generated from the LL data representation. As previously mentioned, the first two dimensions, i.e., h and v, index the SAI location using horizontal and vertical coordinates, and the remaining two dimensions index the x and y spatial position within each SAI, as shown in Figure 2.12.

The 4D LF data representation shares the initial steps with the LL data representation, where demosaicking and devignetting pre-processing steps are applied. After these initial steps, LL to SAIs conversion is applied in two steps. In the first step, resampling, rotation and scaling is applied, such that all MI centers become aligned with an integer grid of pixels. The second step applies slicing to the LF image with the aligned MI centers, therefore splitting the LF image into identically sized rectangles, effectively converting the hexagonally sampled data into a square based grid of MIs. After this step, the SAIs can be constructed by extracting one pixel from a

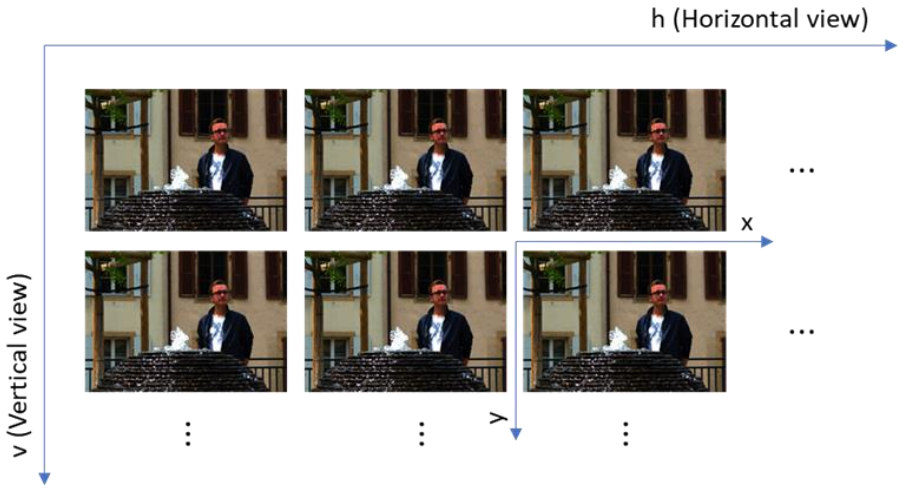


Figure 2.12. 4D LF data representation indexing (image "Fountain and Vincent 2", i.e., I09 from the EPFL LF dataset) [22].

fixed position in each MI and organizing them into a matrix. When using the Lytro Illum, 15×15 SAIs with an individual resolution of 625×434 pixels can be obtained from the LF image. This means that the total number of pixels is increased by about 47%, when compared to the LL data representation.

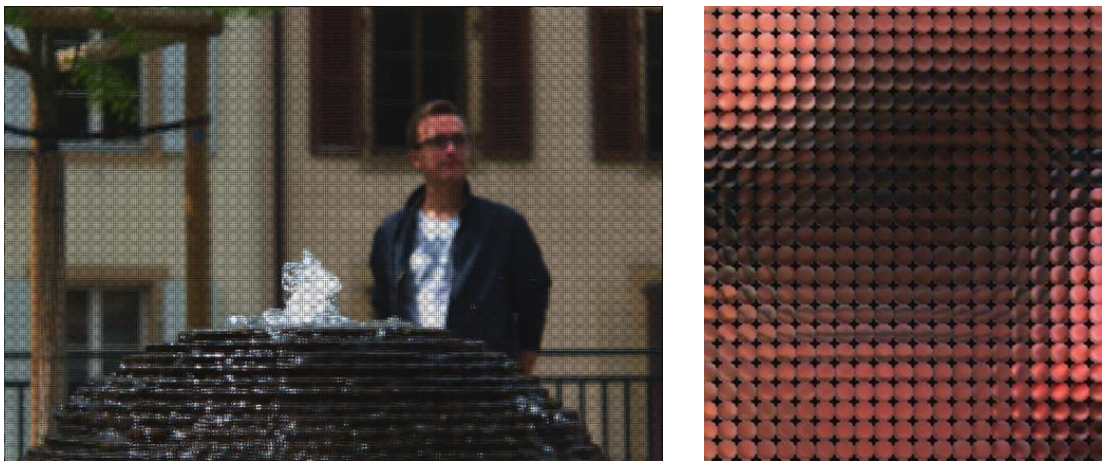


Figure 2.13. Full LF image (left) and zoom of the same LF image (right), using the 4D LF data representation.

Once the LF image is converted into the 4D LF data representation, i.e., a stack of SAIs, it can be organized in several ways. For example, a single LF image with concatenated squared 15×15 pixel MIs (4DLF-MI) as in Figure 2.13; or generating a so called pseudo video sequence (PVS), using the SAIs where a scanning order like raster or spiral is used (4DLF-PVS). All these

conversions are reversible due to the four-dimensional indexing, which also facilitates view rendering.

2.3. Light field encoding

The LF technology has recently attracted many research groups and also standardization bodies, such as JPEG and MPEG. This occurs not only because of its new appealing features and the necessity to normalize the LF data representation, but also due to the very large amount of data associated with these LFs. Thus, these groups are currently developing coding standards for emerging imaging technologies like LF, point cloud, holographic and 360º-video content, whose activities are known as JPEG Pleno [7] and MPEG-I [8], which structure is shown in Table 2.2.

Table 2.2. Structure of both MPEG-I ISO/IEC 23090 (left) and JPEG Pleno ISO/IEC 21794 (right).

MPEG-I ISO/IEC 23090 [23]	**JPEG Pleno ISO/IEC 21794** [24]
Part 1: Immersive Media Architectures	Part 1: Framework
Part 2: **Omnidirectional Media** Format	Part 2: **Light Field** Coding
Part 3 Versatile Video Coding	Part 3: Conformance Testing
Part 4: Immersive Audio Coding	Part 4: Reference Software
Part 5 **Point Cloud** Compression	Part X: **Point Cloud** Coding
Part 6: Immersive Media Metrics	Part Y: **Hologram** Coding
Part 7: Immersive Media Metadata	Part Z: Quality Assessment
Part 8: Network Based Media Processing	
Part 9: **Geometry-based Point Cloud** Compression	
Part 10: Carriage of Point Cloud Data	
Part 11: Implementation Guidelines for Network-based Media Processing	
Part 12: Immersive Video	

The emerging imaging technologies that each standard is tackling are also highlighted in Table 2.2 in bold. The scope of this Thesis is directly aligned with JPEG Pleno Part 2: Light Field, which, therefore, will be reviewed more thoroughly in Chapter 3, while some examples of

functionalities of both MPEG-I and JPEG Pleno will be briefly discussed in the following sections.

2.3.1. MPEG-I functionalities

Although MPEG-I does not directly consider 4D LF signals, it is structured to have several functionalities and use cases that can be considered specific cases of LF applications. The following functionalities [25] are possible by using immersive media such as 360º video and point clouds:

- **AR/VR functionalities:** Support for one or multiple users in an AR/VR environment, e.g., telepresence; multiple user embodiment in a 360 video; object interaction in AR/VR;
- **3DoF/3DoF+/6DoF capabilities:** When wearing a head mounted display (HMD), the user is allowed different types of movement freedom, depending on the number of DoF that are supported. Figure 2.14 shows the various possible movements according to the available DoF [26].

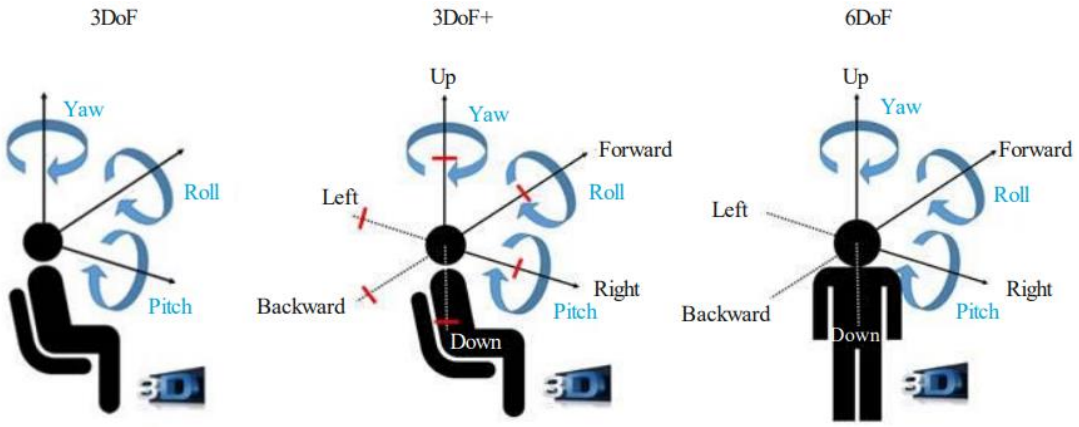


Figure 2.14. Different types of movement freedom that are allowed for 3DoF, 3DoF+ and 6DoF [26].

- **2D/3D support:** The users can experience both monoscopic and stereoscopic video content, depending if one or two viewpoints are being rendered and displayed for each user.
- **Free-viewpoint television (FTV):** The user is able to freely select the monoscopic or stereoscopic 360º content from multiple viewpoints of the same scene.

2.3.2. JPEG Pleno functionalities

JPEG Pleno has decided to put special emphasis on three major plenoptic modalities, notably light fields, point clouds and holography, as well as adding the possibility to mutually convert from one modality to another. However, currently (at the time of writing) the identification of use cases and definition of associated requirements is still ongoing for both point cloud and holographic modalities [24]. Regardless, several functionalities are being considered for each plenoptic modality:

- **Holography:** Despite the limited success in applications and market, specifically with analogic holograms, several applications are proposed by JPEG Pleno, which include microscopy and interferometry [24].
- **Point clouds:** The same functionalities that are reported by MPEG-I can be applied in this case, such as telepresence or FTV.
- **Light field:** Due to the flexibility of the 4D nature of the LF image, it can be used for a number of scenarios which range from capturing and representing a 2D, stereo and multiview scenes, using different capturing devices, as well as the ability to refocus the rendered image after the picture has been taken.

2.4. Light field rendering

After the LF image has been encoded and decoded, depending on the LF capturing device and representation, it might be necessary to render one or several viewpoints from the LF content:

- **2D camera array/2D camera gantry**: When the LF is captured by such types of capturing device, the rendering step is not necessary, if the desired viewpoint corresponds to the position of each 2D camera. This happens because each viewpoint is already captured individually, i.e., a standard 2D image or video, therefore the content can be displayed as is.
- **Unfocused LF camera**: When the unfocused LF camera is used to capture a scene, a lenslet LF is captured, which requires a rendering step in order to generate the individual viewpoints, or SAIs. This step can be done before or after the encoding step, depending on the application and chosen LF representation. For example, when a SAI-based

representation is used, the viewpoint rendering needs to be applied before the encoding step. Regardless, the process that generates SAIs from the LL LF image consists in extracting one pixel from each MI in a fixed position and organizing those pixels in a matrix, forming a SAI [5]. The pixel location that is chosen, determines the SAI that is being selected. In order to perform this viewpoint rendering it is necessary to know which pixel in each MI corresponds to the central pixel, so the desired SAI can be chosen relatively to the central viewpoint. Such information can be normally inferred from metadata that is captured with the LF image, i.e., the white images referred in Section 2.2.1. This process is already performed when using the 4D LF representation. Refocusing can be selected by applying a shear transform in the 4D LF representation [27]. Depending on the angle that is used during the transform step different focus planes can be selected.

- **Focused LF camera**: The requirements in terms of LF rendering for the focused LF camera are the same as the unfocused LF camera. The main difference is the fact that instead extracting one pixel per MI, a patch of pixels is extracted to compose the desired SAI. The size of the patch determines the focus plane that is being selected. When larger patches are used, this means that the selected focus plane is closer to the camera than when using smaller patches. In this case, since more than one pixel is extracted from each MI, the spatial resolution is higher than when the unfocused LF camera is used, however a smaller number of SAIs can be generated [6]. The patches that are combined to generate the SAI can be either directly concatenated, i.e., basic rendering, or they can be combined using a blending algorithm using a slightly larger patch, i.e., blended rendering. Blended rendering is a rendering method that tries to mitigate the visual artifacts caused by basic rendering due to non-matching patches in areas of the SAI that are not in focus [6].

Additionally, regardless of the capturing device, it is possible to synthesize virtual viewpoints from existing ones through the depth image-based rendering (DIBR) process. In such case, depth information is necessary to generate the virtual viewpoints, which can be either transmitted as side information during the encoding and decoding process or it can be generated from the existing viewpoints [28].

2.5. Light field display

LF images, as 4D signals, can be displayed and experienced by the end user in various ways. Although the main goal of this technology is to be able to display the full-fledged 4D LF in a full parallax, glasses-free display, several formats can be derived from the full 4D LF image. These formats can be displayed in other types of display devices, such as:

- **Standard 2D displays**: Multiple 2D images can be extracted from a LF image, which allows the user to visualize regular 2D images.
- **Stereo 3D displays**: A pair of 2D images is rendered from a LF image, which can be visualized in a stereo 3D display. In order to correctly visualize the 3D content (depth perception), normally, glasses are required to separate the left and right view that is displayed to each corresponding eye.
- **Multiview autostereoscopic displays**: When multiple pairs of images are rendered from the 4D LF image, the user is able to visualize several horizontal viewpoints (horizontal parallax) as opposed to the stereo 3D display where just a fixed pair of viewpoints is presented to the user. This technology allows, therefore, a more realistic depth illusion when compared to the stereo 3D display.

It is also worth mentioning that even without a 4D LF display, all the legacy display types, can be used to display LF by allowing functionalities like FTV. In this case, the end user is able to select the desired point of view of the scene, regardless of how many viewpoints the display is able to reproduce at the same time.

The advances in display technology have also allowed to develop LF displays prototypes. Some examples of prototypes are shown in Figure 2.15, namely the FOVI3D LF display [29] and the HoloVizio 80WLT LF display [30].

The FOVI3D LF display [29] uses the same principle as the MLA-based capturing technology. However, instead of using an MLA to sample the angle of the LF hitting the sensor, a flat panel with an MLA overlay is used. The light rays are intersected by the MLA recreating the LF that was captured through the inverse process. The technology used by the Holovizio 80WLT LF display [30] is the so called super-multiview LF display which uses a very dense number of views to replicate the 4D LF. Since this display technologies are able to reproduce the light,

Figure 2.15. FOVI3D LF display [29] (left) Holovizio 80WLT LF display [30] (right).

both in the spatial and angular domain, as it was observed during the content acquisition, it allows for a full parallax (horizontal and vertical) 3D experience without the need for glasses.

Finally, another display technology that has become more affordable, while providing a more natural and immersive experience when compared to 3D displays, is the HMD. HMDs are essentially capable of reproducing either AR or VR experiences by generating several virtual viewpoints of a 3D scene, which are shown according to the user head movement. There are several examples of HMDs in the market that are already able to create very realistic full parallax 3D experiences, such as the Oculus Quest [31] or the Valve Index [32] (shown in Figure 2.16). Moreover, in Figure 2.17 another variant of HMDs [33] is shown, which instead of using regular 2D displays such as the previously mentioned HMDs it uses small LF displays that take advantage of the same MLA-based principle explained before.

Figure 2.16. HMDs examples, Oculus Quest [31] (left) and Valve Index [32] (right).

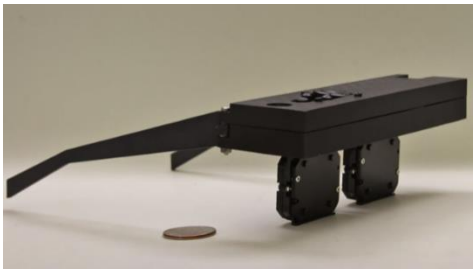


Figure 2.17. NVIDIA Near-Eye head mounted display [33].

2.6. Light field objective evaluation

JPEG Pleno also contributes for establishing the objective evaluation tools to compare different LF image codecs. In order to perform LF objective evaluation during this research, two processing chains were used, according to the provided metadata and type of camera used to acquire the LF:

- **Base evaluation processing chain (BEPC):** This processing chain is used whenever the acquired LF image does not have any metadata to determine the center of each MI, i.e., making it difficult to correctly render SAIs. The objective evaluation is performed using the LL representation, i.e., evaluating the captured LF directly. LF images from any type of LF camera, i.e., unfocused or focused, can be compared using this processing chain, however, it is more prominently used with focused LF cameras.
- **JPEG Pleno common test conditions (CTCs):** This processing chain is based on generating and comparing a LF image in the SAI domain using the 4D LF data representation. This processing chain is used with LFs captured with non-MLA LF capturing devices and unfocused LF cameras. When evaluating focused LF cameras images, this processing chain is not used because when applying SAI rendering it uses a patch size of one pixel, i.e., effectively only rendering SAIs which are focused on furthest possible focus plane.

2.6.1. Base evaluation processing chain

The BEPC is used to perform objective quality evaluation without using any metadata, which processing chain is shown in Figure 2.18.

Figure 2.18. Base evaluation processing chain.

As can be seen in this figure, after the LF image is acquired with any form of MLA based LF capturing device, only demosaicking is applied prior to directly encoding the LL LF image. After LF coding and decoding, the output LF will have the same data representation, i.e., LL. This output LF can then be compared against the reference LF image, which is generated by performing all the steps in the base evaluation processing chain, with the exception of the encoding/decoding steps, i.e., effectively only measuring the impact of the coding process.

The objective quality impact of the coding process, $PSNR_Y$, between the output LF (LF_O) and the reference LF (LF_R) is calculated as defined by (2.2):

$$PSNR_Y = 10 \log_{10} \frac{MAX^2}{MSE_Y} \tag{2.2}$$

where MAX is $2^n - 1$, i.e., n is the bit depth. For example, for $n = 8$, $MAX = 255$. The mean square error (MSE_Y) is calculated as defined by (2.3):

$$MSE_Y = \frac{1}{WH} \sum_{y=0}^{H-1} \sum_{x=0}^{W-1} [LF_O(y,x) - LF_R(y,x)]^2 \tag{2.3}$$

where W and H are the width and height, respectively, of the LF image in the LL data representation.

2.6.2. JPEG Pleno common test conditions

The JPEG Pleno CTCs are used to perform objective quality evaluation on LF images that were captured using for both unfocused LF cameras and non-MLA based solutions, such as HDCAs, with the processing chain shown in Figure 2.19.

The main difference between the JPEG Pleno CTCs processing chain for evaluating the LF objective quality and the BEPC, is the fact that, in the case of the JPEG Pleno CTCs processing chain, the objective quality is calculated in the SAI domain using the 4D LF data representation.

Figure 2.19. JPEG Pleno CTCs processing chain.

The LF toolbox [34] is recommended when using the JPEG Pleno CTCs processing chain, which is responsible for all the pre-processing steps, namely, demosaicking and devignetting, conversion to 4D LF data representation and color and gamma correction. After the pre-processing steps are applied, the 4D LF image can be encoded using one of the mentioned 4D LF data representation variants shown in Section 2.2. After the decoding step, the output LF image will in the same 4D LF data representation, so it can be compared in the SAI domain. As in the base evaluation processing chain, the reference LF is created by using the same processing chain as the output LF, i.e., using the processing chain in Figure 2.19, with the exception of the LF encoding/decoding blocks.

The number of SAIs that will be generated depends on the angular resolution of the LF capturing device. In order to perform objective evaluation, the PSNR of each SAI color component is calculated as defined by (2.4):

$$PSNR_{Y,U,V}(v,h) = 10\log_{10}\frac{MAX^2}{MSE_{Y,U,V}(v,h)} \tag{2.4}$$

where the $MSE_{Y,U,V}$ of each SAI is calculated as defined by (2.5):

$$MSE_{Y,U,V}(v,h) = \frac{1}{WH}\sum_{y=0}^{H-1}\sum_{x=0}^{W-1}[LF_{O_{Y,U,V}}(v,h,y,x) - LF_{R_{Y,U,V}}(v,h,y,x)]^2 \tag{2.5}$$

where LF_O is the output LF and LF_R is the reference LF. The (v,h) coordinates are used to access the desired output SAI. For example, the top left SAI can be accessed by using $(v,h) = (0,0)$.

In order to combine the contributions of the PSNR from each color component, the $PSNR_{YUV}$ of each individual SAI can be calculated as [35], as shown in (2.6):

$$PSNR_{YUV}(v,h) = \frac{6PSNR_Y(v,h) + PSNR_U(v,h) + PSNR_V(v,h)}{8} \tag{2.6}$$

The average $PSNR_{YUV}$ and $PSNR_Y$ is normally used to perform objective evaluation, however, the number of SAIs used to calculate the average depend on the capturing device. Normally, all the SAIs are considered for the average calculation, with the exception of the LF images captured by Lytro Illum [20] cameras. In the case of the Lytro Illum, the 4D LF generated from the LF Toolbox is composed by 15×15 SAIs [34]. However, since the outer SAIs are mostly unusable they are not considered for the average calculation [35]. As such, only the inner 13×13 SAIs are considered. The average $PSNR_{YUV}$ value can be computed as defined by (2.7):

$$PSNR_{YUV_{mean}} = \frac{1}{(V-2)(H-2)} \sum_{v=1}^{V-2} \sum_{x=1}^{W-2} PSNR_{YUV}(v,h) \tag{2.7}$$

where V and H are the maximum number of vertical and horizontal SAIs, i.e., 15. Consequently, the average $PSNR_Y$ can also be computed as defined by (2.8):

$$PSNR_{Y_{mean}} = \frac{1}{(V-2)(H-2)} \sum_{v=1}^{V-2} \sum_{x=1}^{W-2} PSNR_{Y}(v,h) \tag{2.8}$$

2.7. Final remarks

This chapter presented the LF imaging principles of the LF storage/transmission processing chain, which ranges from:

- **LF image acquisition**: The LF can be acquired through conventional 2D cameras or using LF cameras based on MLAs.
- **LF representation**: The most common LF data representations include the "Raw" Lenslet and 4D LF formats.
- **LF encoding/decoding**: The JPEG Pleno: Part 2 standard is specifically designed with LF in mind while MPEG-I standard allows for other types of media (specific cases of LF), such as 360-video, to be encoded.
- **LF rendering**: The rendering step depends on the type of capturing device that was used, i.e., typically, only MLA-based LFs requires a rendering step, unless the desired view is a virtual view.

- **LF display**: The LF content can be visualized from a vast range of solutions: from regular 2D displays up to full-fledged 4D LF displays.
- **LF objective quality evaluation**: Two objective quality evaluation techniques were presented which are used depending on the provided metadata and LF acquisition type.

Chapter 3. Related work on light field image coding

The LF transmission pipeline requires that all the traditional processing blocks present in traditional image storage/transmission systems to be adapted to the 4D nature of the LF signal. Although the LF image can be represented as a 2D signal and consequently to be encoded and decoded by a standard 2D image or video coding standard such as HEVC [36], it is not expected that such coding solutions will fully exploit the redundancy and intrinsic characteristic of the LF signal. Therefore, the additional degrees of freedom that exist in the LF technology need to be considered when developing a specific LF coding solution. As such, it is useful to first identify the four types of redundancy that can be exploited when encoding a LF signal, which are intrinsically connected to the representation that is being used, i.e., MI redundancy or SAI redundancy:

- **MI redundancy:**
    1. Intra-MI redundancy that exists between neighboring pixels within an MI.
    2. Inter-MI redundancy that exploits the similarity between different MIs, which is also known as non-local spatial redundancy.
- **SAI redundancy:**
    1. Intra-SAI redundancy that exists between neighboring pixels within a SAI, which can be related with conventional spatial redundancy in image and video coding standards;
    2. Inter-SAI redundancy, that exploits the similarity between different SAIs, corresponding to different perspectives of the same scene. This can also be referred to as inter-view redundancy in multiview image and video coding standards;

In this context, several solutions have been proposed in the literature, which can be categorized by the type of exploited redundancy: MI-based approaches or SAI-based approaches. Alternatively, there are some methods which do not exploit directly the redundancy of either SAIs or MIs, using instead alternative representations to encode the LF signal, such as low rank approximations.

The remainder of this chapter is organized as follows: Section 3.1 describes a summary of the coding tools present in HEVC standard, which are relevant due to their use in several state-of-the-art methods, as well as in several contributions of this Thesis; Section 3.2 presents a summary of the coding tools defined in JPEG Pleno Part 2: LF coding; Section 3.3 to 3.5 describe several state-of-the-art contributions on LF image coding, which rely on exploiting redundancy from MIs, SAIs or alternative representations, respectively; finally Section 3.6 concludes this chapter with some final remarks.

3.1. High efficiency video coding standard

The main concept of the HEVC standard [36] coding architecture, shown in Figure 3.1, is similar to its predecessor H.264/AVC [37]. Each picture is split into blocks, which are predicted in the spatial or temporal domains. Depending on the prediction scheme, each picture can be encoded using I, P or B modes. In I-pictures, intra prediction is applied by exploiting only spatial redundancy. In P- or B-pictures, temporal prediction can be additionally used. The residue, i.e., the difference between the predicted block and the original block, is encoded using a spatial transform. The resulting coefficients are scaled and quantized. This information is entropy coded and transmitted with the prediction information, like the selected mode or the motion vectors (if applicable). The encoder also includes a decoder, such that the decoding process is replicated in the encoder. This allows the decoded pictures to be used as reference pictures for prediction of the future frames. The decoding loop includes the reconstruction of the approximated residue using inverse quantization and transform. The residue is added to the prediction generating the decoded block. In order to reduce coding artifacts, the decoded picture may be filtered after or during the decoding process.

The HEVC standard only specifies the bitstream structure and syntax, as has been the case for all past ITU-T and ISO/IEC video coding standards. This way, the standard provides maximum

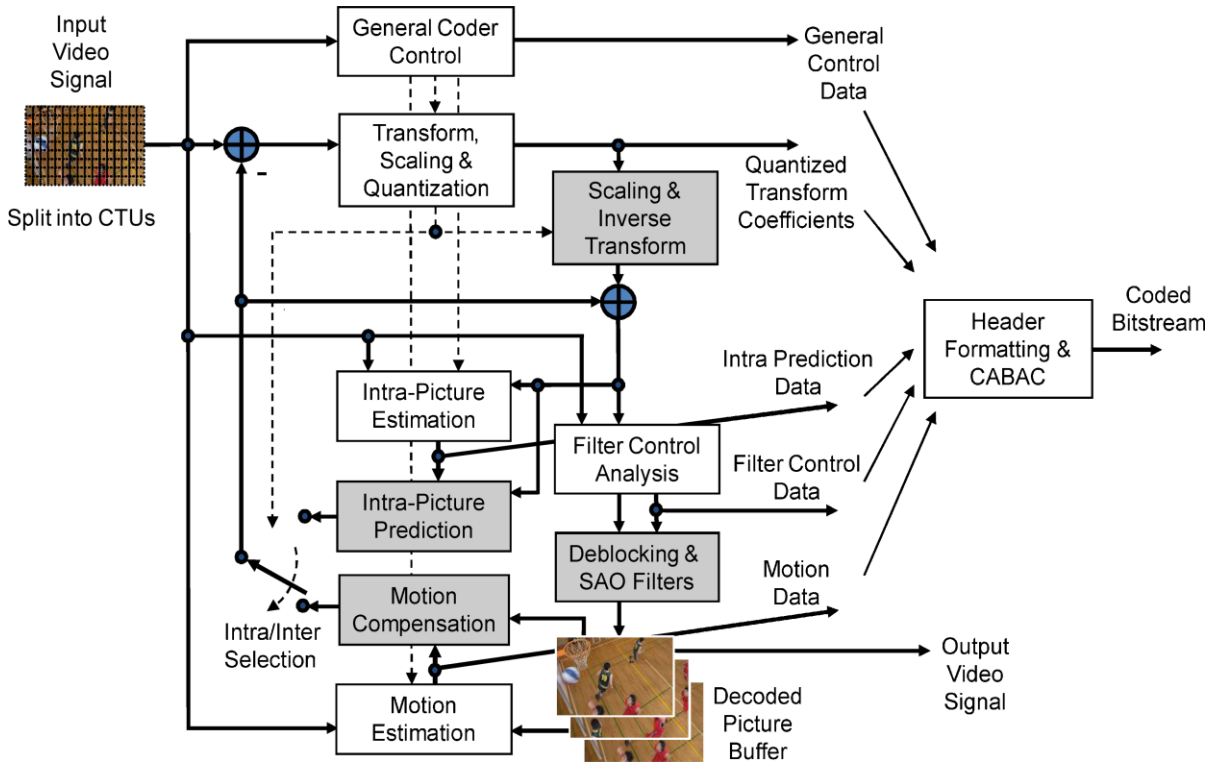


Figure 3.1. HEVC video encoder architecture [36].

flexibility to optimize implementations on the encoder side while still being standard complaint. The following sections describe in more detail the most important blocks within HEVC, which include block partitioning, intra and inter prediction, spatial transform, entropy coding and in-loop filters, since these blocks can be extremely relevant for LF encoding.

3.1.1. Block partitioning

The input video signal is split into coding tree units (CTU) which are quadtree elements. CTUs are composed by coding tree blocks (CTB), one for the luminance and two for the chrominance components. The possible luminance CTB sizes are 64×64, 32×32 and 16×16. The CTUs are divided into coding units (CU), which contain luminance coding blocks (CB) and two chrominance CBs. CUs can have prediction units (PU) and transform units (TU). A PU is a unit that can have a size between 4×4 and 64×64, where either intra or inter prediction is performed at a CU level. TU size depends on the PU size, varying from to 32×32 down to 4×4. Each CU may have more than one PU and it is always square shaped, with a size between 8×8 and the size of the CTU. A CB with $M \times M$ pixels can have partitioned prediction blocks (PB), which are the luminance and chrominance components of a PU, as $2N \times 2N$, $N \times N$, $N \times 2N$ and

$2N \times N$, where $M = 2N$. A PB can also be partitioned asymmetrically using asymmetric motion prediction (AMP) using sizes: $2N \times nU$, $2N \times nD$, $nL \times 2N$ and $nR \times 2N$, where U, D, L and R correspond to Up, Down, Left and Right. For example, if $M = 16$, the $2N \times nU$ the resulting partition will be one 16×4 block on top and a 16×12 on the bottom, because the division is horizontal and on the upper part of the block.

All possible sizes for CTU, CU, TU and PU are shown in Table 3.1 in a hierarchical order. For example, for 16×16 CTU size, the only possible sizes for CU, PU and TU are the ones on the same line or below in the table. Therefore, the only possible CU sizes are 16×16 and 8×8 and TU sizes are between 16×16 and 4×4.

Table 3.1. Possible CTU, CU, TU and PU sizes.

M	N	CTU	CU	TU	PU	
64	32	64×64	64×64	-	64×64	2N×2N
					64×32	2N×N
					32×64	N×2N
					64×16 / 64×48	2N×nU / 2N×nD
					16×64 / 48×64	nL×2N / nR×2N
32	16	32×32	32×32	32×32	32×32	2N×2N
					32×16	2N×N
					16×32	N×2N
					32×8 / 32×24	2N×nU / 2N×nD
					8×32 / 24×32	nL×2N / nR×2N
16	8	16×16	16×16	16×16	16×16	2N×2N
					16×8	2N×N
					8×16	N×2N
					16×4 / 16×12	2N×nU / 2N×nD
					4×16 / 12×16	nL×2N / nR×2N
-	-	-	8×8	8×8	8×8	N×N
					8×4	N×N/2
					4×8	N/2×N
-	-	-	-	4×4	4×4	N/2×N/2

3.1.2. Intra prediction

The similarities between neighboring pixels in homogenous regions, patterns or contours in 2D images are normally referred to as spatial redundancy. The concept of spatial redundancy can be exploited by image/video codecs to improve the compression efficiency.

In HEVC, spatial redundancy is exploited by using 35 intra prediction modes, which use decoded boundary samples of adjacent blocks as reference data to predict the current PU. The

intra predictions modes include: DC mode, planar mode and 33 directional modes. The DC mode creates a flat surface that is the mean of the boundary samples, which is useful when trying to predict homogeneous regions. The planar mode is efficient when trying to predict gradient surfaces of the picture. The directional modes are very efficient in predicting image contours, which is also the reason the existence of 33 different directional modes. Each directional mode represents a different direction that can be exploited. These directional modes are used to predict contour directions which range from 45º (directional mode 2) to -135º (directional mode 34), as shown in Figure 3.2.

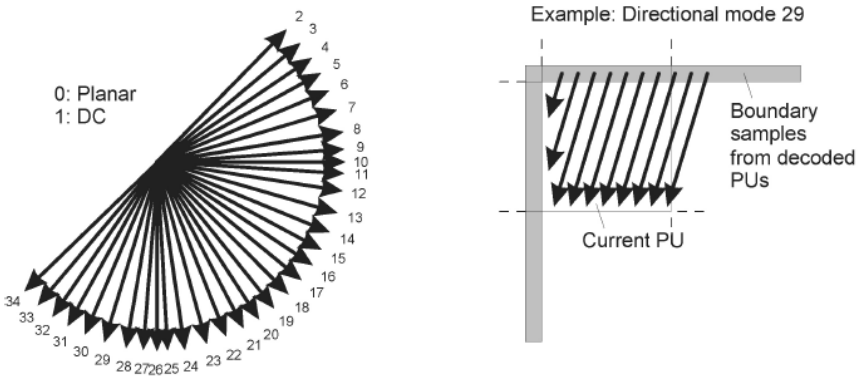


Figure 3.2. Modes and directional orientations for intra prediction modes [36].

Depending on the chosen direction, different boundary samples from decoded PUs are used to predict the current PU. All directional modes are available for PU sizes of 32×32, 16×16 and 8×8. However, when the PU is 64×64 only four directional modes are available and if the PU is 4×4 only sixteen are available.

3.1.3. Inter prediction

Successive frames in a video sequence normally exhibit a large amount of similarities depending on the scene's motion, i.e., a static scene will have much more temporal redundancy than an action scene, for instance.

HEVC includes several inter prediction modes that can be used to exploit the temporal redundancy. When encoding the current PB, one or several decoded picture references will be available in the reference picture lists (RPL), which will be used by HEVC to describe the motion of the current PB using a 2D displacement vector. The way this motion vector is calculated and transmitted to the decoder depends on the used inter prediction mode:

- **Merge:** The motion vector is derived from a spatial or temporal neighboring PB, and no residual data is encoded. This is typically used when the current PB has the same motion vector as a neighboring PU. Figure 3.3 shows the several spatial candidates that can be used to infer the motion vector. The availability of each candidate is checked in the following order $\{a_1, b_1, b_0, a_0, b_2\}$. The temporal candidates include the right bottom position, and if not available, the center position, i.e., a motion vector from a co-collocated PB. The skip mode is a special case of merge mode that is used when the motion vector is derived from the first available candidate.
- **Motion estimation**: The motion vector is determined using a (non-normative) matching algorithm to identify the PB on previously encoded pictures that correspond (best match) to the current PB of the current picture. The motion is therefore identified explicitly by a motion vector that has one quarter pixel precision. HEVC uses an eight-tap filter to interpolate the half-sample positions and then a seven-tap filter for the quarter-sample positions. After the motion vector is determined, advanced motion vector prediction (AMVP) is used, so the motion vector can be transmitted relatively to previously encoded motion vectors. AMVP uses the same candidates shown in Figure 3.3 as predictors for the determined motion vector.

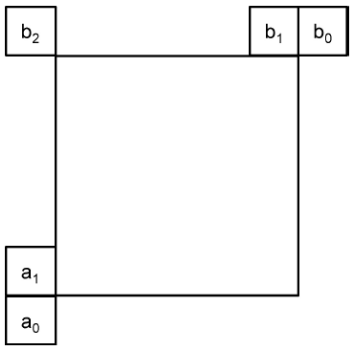


Figure 3.3. Positions of spatial candidates for merge and motion estimation inter prediction modes [36].

The motion vector can be generated relatively to one or two reference pictures. In the second case two vectors are generated for two different reference pictures; the two PBs generated by this process (bi-prediction) are combined by the use of a simple average or an explicit weighted average.

3.1.4. Spatial transform

After applying the intra or inter prediction modes to the PB, the residue block is generated, by subtracting the original block with the PB. This residue block is encoded by HEVC and transmitted to the decoder, however, unlike the prediction information, the residual information is lossy encoded. The discrete cosine transform (DCT) is used to encode the residual block by converting it into the frequency domain allowing its energy to be concentrated in a small number of DCT coefficients. The DCT coefficients can be further compressed by applying quantization as in H.264/AVC, but the DCT block sizes can vary from 32×32 down to 4×4. The higher the QP, the lower the quality of the reconstructed residual block.

HEVC also uses an alternative transform for 4×4 TBs, the discrete sine transform (DST), providing marginal better results than the DCT when the residue amplitude is higher on the pixels that are furthest way from the boundary samples used for intra prediction.

3.1.5. Entropy coding

Entropy coding is used to encode all the symbols generated by the encoder, which include quantized transform coefficients, intra prediction data, motion data and filter control data. Entropy coding exploits the statistical redundancy of the occurrence of each symbol. The context adaptive binary arithmetic coding (CABAC) core algorithm remains unchanged from H.264/AVC. However, improvements have been made in terms of computational complexity and memory requirements. Its output is the HEVC bitstream.

3.1.6. In-loop filters

One of the main drawbacks of block-based coding is the generation of blocking artifacts, i.e., visible discontinuities at the block boundaries, on the decoded image, especially when encoding an image/video with a high QP. In order to mitigate this problem two filters are applied by the decoder to the reconstructed image:

- **Deblocking filter (DBF)**: It was designed to improve the subjective quality of the decoded video by reducing blocking artifacts as shown in Figure 3.4. This filter is applied to all samples of adjacent PU or TU boundary using three filter intensity options, that are chosen by HEVC depending on the content.

Figure 3.4. Reconstructed image before DBF (left) and after DBF (right) [38].

- **Sample adaptive offset (SAO)**: It is used after DBF in order to further improve the signal representation in smooth areas and around edges as shown in Figure 3.5. The application of SAO consists in adding an offset value to the decoded samples that is based on a look-up table transmitted by the HEVC encoder.

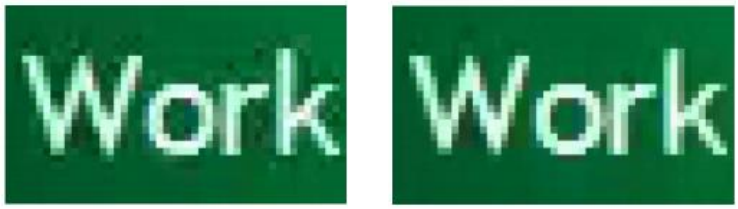


Figure 3.5. Reconstructed image before SAO filter (left) and after SAO filter (right) [39].

3.2. JPEG Pleno part 2: light field coding

The JPEG Pleno part 2 describes the LF coding part of the upcoming standard. JPEG Pleno part 2 is divided into several annexes describing each structural part of the standard. It provides two separate solutions to encode a LF image [40] which are described as 4D Transform Mode, i.e., Annex B, and 4D Prediction Mode, Annexes C, D and E. These solutions correspond to adaptations of the state-of-the-art methods Multidimensional LF encoder (MuLE) [41] and hierarchical warping, merging and sparse prediction encoder, (WaSP) [42]. Two solutions were proposed because they provide more efficient results for different types of LF images, where MuLE is more efficient for MLA-based LF images and WaSP is more efficient for LF captured with HDCAs. Such solutions are described in the following sections.

3.2.1. MuLE: Multidimensional light field encoder

MuLE [41] LF coding solution was developed with the 4D LF representation in mind. Instead of focusing exclusively on either MI- or SAI-based redundancy exploitation, it is focused on the full 4D redundancy of the LF image. It works in three steps:

1. **4D transform:** A 4D-DCT, separable spatial transform, is applied to a LF image in each one of the four dimensions, i.e., both spatial (x, y) and angular dimensions (h, v), thus exploiting the redundancy in each dimension. The generated coefficients are grouped into a 4D array of subbands of the same frequency transform coefficients. As in most 2D coding scenarios it is expected that after quantizing most of the energy of the DCT coefficients is concentrated in the lower frequencies, which also applies in this case. These DCT coefficients are then sliced into 4D bit planes, i.e., a binary representation of the 4D DCT coefficients.
2. **Hexadeca-tree bitplane clustering:** The DCT coefficients in the 4D bit planes are clustered using a hexadeca-tree. The concept is similar to a quadtree, however, since in this case, the DCT coefficients are organized in 4D, this means that when each node is segmented, i.e., segmentation flag is true, it has sixteen leaf nodes instead of four, as shown in Figure 3.6. The hexadeca-tree is described using a series of ternary flags signaling:
    i. **Lower bitplane:** This indicates that the decedent of this node is a block with the same dimensions as the original one, however it is represented with a bit plane with a reduced precision of one bit.
    ii. **Split block:** This indicates that this node has sixteen leaf nodes, each one with half the length of the original one in all four dimensions
    iii. **Zero block:** This indicates that this node has no descendants and it is represented by zeros.

    Each leaf node is going to be further partitioned until the 4D block is $1 \times 1 \times 1 \times 1$ or if the coefficient magnitudes are below a certain threshold.

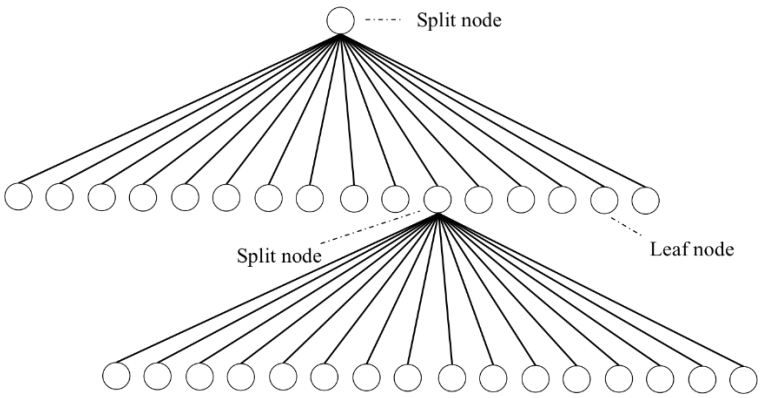


Figure 3.6. Hexadeca-tree representing the clustering of bit-planes of 4D transform coefficients.

3. **Entropy coding:** When the clustering step is concluded, a context-based binary adaptive arithmetic coder is used to encode both the segmentation flags and the DC and AC coefficients per bitplane. There are three different contexts for each one of the respective types of information being encoded. The output of the entropy coder is the MuLE bitstream.

3.2.2. WaSP: Hierarchical warping, merging, and sparse prediction

WaSP [42] LF coding solution was designed to be an extension of JPEG 2000 allowing backward compatibility within the JPEG standards. This hierarchical approach exploits the SAI-based redundancy which is present in the LF image. WaSP is used to hierarchically encode a LF image using four different steps:

1. **Encode the reference SAIs:** N reference SAIs at the lowest hierarchy level are encoded independently using JPEG 2000.
2. **Encode reference inverse depth maps:** At least one reference SAI inverse depths maps, i.e., $1/D$, where D is the depth of the SAI, at the lowest hierarchy level are encoded using JPEG 2000.
3. **Perform warping merging and sparse prediction for each non-reference SAI:**
    i. **Warping:** Both the reference inverse depth and reference SAIs are warped to the position of the target SAIs, i.e., all the remaining non-reference SAIs. After this step, N synthesized versions of both the inverse depth and SAIs are generated for each non-reference position.

ii. **Merging:** The warped synthesized SAIs are merged (combined) using the camera centers and the coefficients of an optimal least square merger. For low bitrates, however, the merging process can be done using fixed weights derived from the distance between the current SAI and warped SAI. Occlusions are mitigated using a post processing median filter. The warped synthesized inverse depth maps use an algorithm that, although not optimal, requires no access to a ground truth inverse depth [42], which is necessary because normally, there is no ground truth available for the SAI depth maps. The merging step information, i.e., the necessary weights to apply to merging step, are transmitted to the decoder.

iii. **Sparse prediction:** The merged SAI is adjusted using a sparse predictor proposed in [43], which improves the quality of the merged SAI by performing an optimal least squares interpolation of neighboring encoded SAIs. The sparse predictor information signaling the neighboring encoded SAIs used for prediction is transmitted to the decoder.

4. **Encode residual of each non-reference SAI:** The improved merged SAI generated in the last step is subtracted to the original non-reference SAI generating the residual SAI, which is then encoded using JPEG 2000.

3.3. MI-based approaches

This section presents the main contributions to efficiently encode LF images which are mostly based on exploiting the MI redundancy. This includes intra- and inter-MI redundancy, which is a characteristic type of redundancy of LF images. This new type of redundancy can typically be exploited using search-based approaches, explained in Section 3.3.1. Alternatively, other approaches based on using spatial transforms, reshaping the MIs or disparity-based approaches can be used to efficiently exploit the MI redundancy, as explained in Sections 3.3.2 to 3.3.4, respectively.

3.3.1. Search algorithm-based approaches

When using the MI-based LF representation, the inter-MI redundancy is normally much higher than the typical image spatial redundancy, therefore, most methods consist on search algorithms that exploit the MI similarity. The search algorithms can be based on different techniques and may use one or multiple references [44]–[55].

- **Unidirectional search:** In [46], a self-similarity (SS) compensated prediction is proposed that takes advantage of the flexible partition patterns used by HEVC. This approach consists in an adaptation of the inter prediction mode present in HEVC, such as, Skip, Merge and Motion Estimation to be used in the causal area, i.e., decoded area, of the LF image as intra modes. Such adaptations are described as SS-Skip, SS-Compensation and SS-Merge. The SS compensated prediction allows the inter-MI redundancy to be exploited by describing the similarities between the current MI and neighboring MIs using 2D displacement vectors, i.e., SS vectors, as illustrated in Figure 3.7. In [44], [45], the SS approach is validated using the H.264 coding standard for both LF image and video coding. Finally, in [50], the SS compensated prediction performance is further improved by adding vector prediction candidates specifically for SS vectors, to be used in both SS-Compensation and SS-Merge.

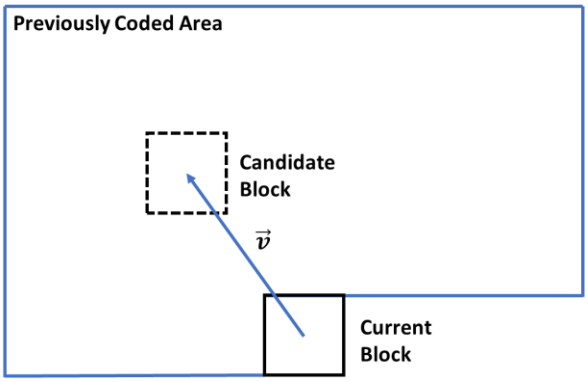


Figure 3.7. Unidirectional search example.

- **Bidirectional search:** The authors in [47] extended the unidirectional search proposed in [46] as a bidirectional search version of SS. This search algorithm can use two displacement vectors from different areas of the LF image, in order to then average the prediction block of both searches, as shown in Figure 3.8. This approach was later tested for LF images captured with an unfocused LF camera [52]. Furthermore it was finally

generalized in [53] as a coding method using up to two hypotheses for prediction in spatial and time domain. Recently, the unidirectional approach proposed in [50] was extended to have weighted bidirectional capabilities, using a locally optimal rate constrained algorithm [55]. Several combinations of weights are tested; however, it is concluded that the RD efficiency is higher when the weights for both directions are closer to 0.5, i.e., regular average.

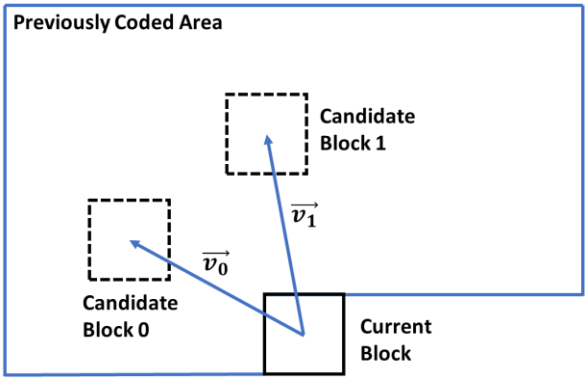


Figure 3.8. Bidirectional search example.

- **Locally linear embedding (LLE):** Alternatively to the several approaches that propose search algorithms capable of combining prediction blocks up to two candidates, the authors in [48] proposed a prediction technique that allows to use *N* directions. As shown in Figure 3.9, a template matching algorithm is applied to find the *N* nearest template candidates of the current block. A least-squares optimization problem is then solved in order to estimate the weights used to linearly combine the *N* template candidates into a prediction block. In [48], this approach was tested using between one and eight neighbor templates. The main advantage of the template matching over the block matching algorithm, as it is used in unidirectional and bidirectional searches, is the fact that the displacement vector information is implicit, i.e., the same template search can be repeated at the decoder in order to generate the same displacement vectors.

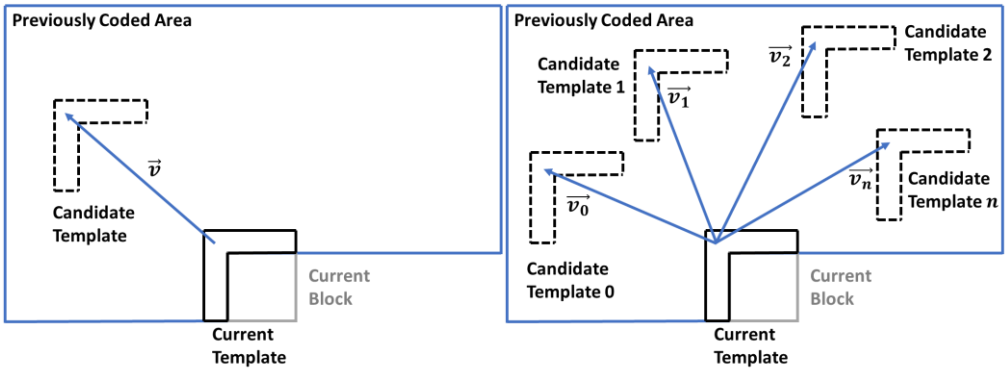


Figure 3.9. Template matching for one (left) and n directions (right).

- **Gaussian process regression:** The authors in [54] proposed an alternative search algorithm that does not require as much computational effort as more traditional search methods, e.g., full search algorithms. A search algorithm similar to [48], however, with a reduced search area, allowing only horizontal and vertical directions, is used to find the *N* nearest neighbor templates in the causal area. This simplified search algorithm is shown in Figure 3.10. The normalized cross correlation (NCC) is used to assess the reliability of the obtained templates. The prediction from the *N* templates is modeled as a non-linear gaussian process and gaussian process regression is used for estimating the prediction block. More recently in order to improve de prediction accuracy for non-homogenous textures and reduce the computational complexity in this work, the authors in [56] proposed to apply a classification method which can segment the non-homogeneous texture areas improving the prediction accuracy. Moreover, the computational complexity is improved by using different prediction modes for each specific area of the lenslet LF image, i.e., content-based prediction.

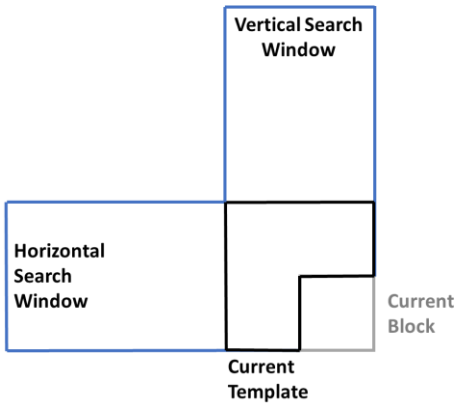


Figure 3.10. Reduced search area proposed in [54].

3.3.2. Transform-based approaches

LF coding schemes can also rely on the use of transforms, e.g., DCT to exploit the intra- and inter-MI redundancy. In [57] a three dimensional version of the DCT is used to exploit the redundancy of a stack of MIs. The 3D-DCT is applied on stacks of four MIs and then the DCT coefficients are quantized, using a 3D scalar quantizer. The resulting quantized coefficients are encoded with a hybrid run-length/Huffman entropy encoder. In [58], this approach was further tested using different stack configuration of MIs, such as $1 \times 2N$ and $2N \times 1$ or $N \times 2$ and $2 \times N$ MIs configurations, where N can be 2 or 4. Figure 3.11 shows how the $1 \times 2N$ configuration is assembled when $N = 2$.

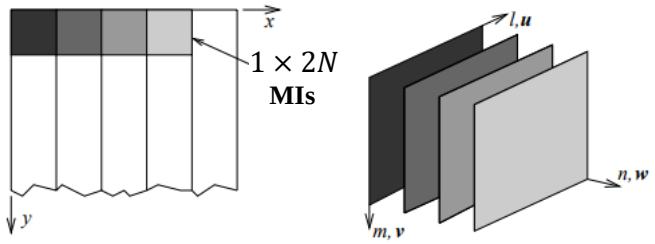


Figure 3.11. Assembling 1×2N configuration for 3D-DCT when N=2 [57].

3.3.3. MI reshaping-based approaches

Most of the MI-based LF coding approaches try to fit a block structure into a lenslet LF image, which is organized in an irregular square or hexagonal grid of MIs, which may not be very efficient. There are two types of alternative approaches that try to fit the LF image structure and the coding structure, which include:

- **Reshaping the LF image structure to fit the coding structure:** In [48, 49], a reversible image reshaping and adaptive interpolation method is designed to align the MI structure with the CTU grid for HEVC, as shown in Figure 3.12. In [59], once the LF image is reshaped, the inter MI redundancy is exploited by using four blocks in four neighboring MIs to create a prediction block for the current MI. In [60], the authors use a prediction mode base on dictionary learning where the MIs are represented by a sparse linear combination of atoms from a generic dictionary.

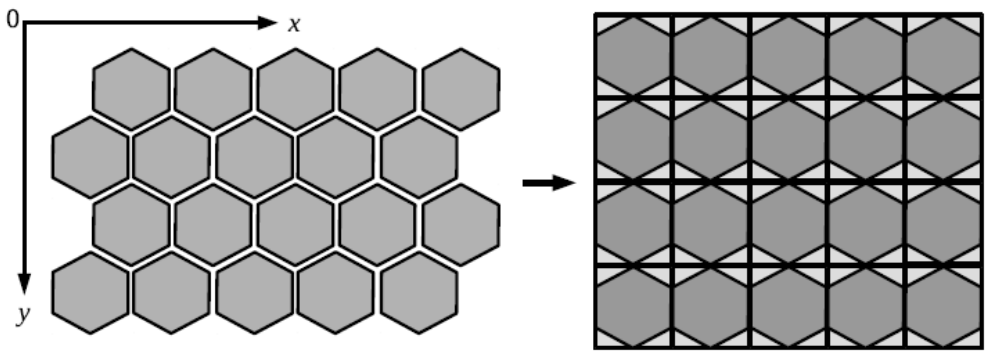


Figure 3.12. LF image before MI reshaping (left) and after MI reshaping (right) [60].

- **Reshape the coding structure to fit the LF image structure:** Alternatively, the authors in [61] propose to change HEVC coding structure in order to recognize an hexagonal MI grid as regular CU structure. In this case, the inter-MI redundancy is exploited using the neighboring MIs with an optimized linear prediction design based on L1 minimization. Figure 3.13 shows the inter-MI prediction mode being applied to the current MI using MIs R_0, R_1 and R_2 as reference MIs. Note that in this approach, each MI is recognized as an individual modified CU by HEVC facilitating the coding procedure.

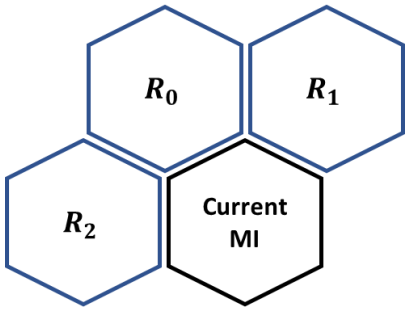


Figure 3.13. MI prediction based on a linear combination of three reference MIs (R_0, R_1 and R_2).

3.3.4. Disparity-based approaches

Since the LF image consist in capturing a scene from several points of view, some authors proposed techniques to exploit the geometrical information of the scene, namely by exploiting the inter-MI redundancy by using the depth information between several MIs. In [49], a unidirectional search algorithm is used, but, instead of applying a full search in order to find the SS vector, the approach considers the estimated disparity between the several MIs. In [62], a prediction mode is proposed that uses a depth-adaptive convolutional neural network (CNN).

Several neighboring blocks in different neighboring MIs feed the CNN that at the output generates a prediction block.

3.4. SAI-based approaches

This section presents the main literature contributions to efficiently encode LF images that are mostly based on exploiting the SAI-based redundancy. This includes intra- and inter-SAI redundancy, i.e., spatial and inter-view redundancy, which is already widely studied in image and multiview codecs, respectively. Since in this case, the most prominent type of redundancy is normally the inter-SAI redundancy, many of the proposed solutions in the state-of-the-art are based either in video codecs or multiview codecs, such as HEVC [36] or MV-HEVC [63], respectively. These types of approaches rely on converting SAIs into a sequence of pictures, i.e., a PVS, or multiple PVS. Since the temporal prediction tools are as effective as inter-view prediction tools in exploiting the inter-SAI redundancy. Alternatively, other approaches based on structural key views (SKV) and spatial transforms are used to efficiently exploit the SAI redundancy. Sections 3.4.1 and 3.4.2 include approaches that rely on PVS and multiview solutions, respectively, while Sections 3.4.3 and 3.4.4 include alternative approaches based on SKVs and spatial transforms, respectively.

3.4.1. PVS-based approaches

Most state-of-the-art video codecs rely on non-normative motion estimation tools, like, for example, block matching algorithms, to exploit the temporal redundancy of video data. As a similar redundancy exists between SAIs, these tools can also be used to exploit the inter-SAI redundancy. To this end, several scanning strategies [64]–[66] have been proposed to transform SAIs into a PVS which is then encoded as a regular video sequence. In [64], [65] two different types of raster scan, i.e., raster and “serpentine” scans, are proposed as well as a spiral scan, which is shown to be in general more efficient than both raster scans. Such scanning orders are illustrated in Figure 3.14. In [66] an improved raster scan is proposed which includes enhancement illumination compensation and an improved reconstruction of the MIs using adaptive filters. In [67], the authors used a PVS scheme to organize the SAIs into layers, depending on the proximity to the central SAI, starting with the central SAI and moving on to the outer SAIs. The further away the SAI is from the center, the higher the value of the used

quantization parameter (QP). More recently, the authors in [68] also proposed 2D hierarchical coding structure with a limited number of reference frames, where the reference frames for each SAI would be selected based on the distance to the current SAI instead of the picture order count (POC), i.e., as in a typical video coding scenario. Additionally, an optimal bit allocation algorithm was proposed for the 2D hierarchical coding structurer which further improves the coding efficiency.

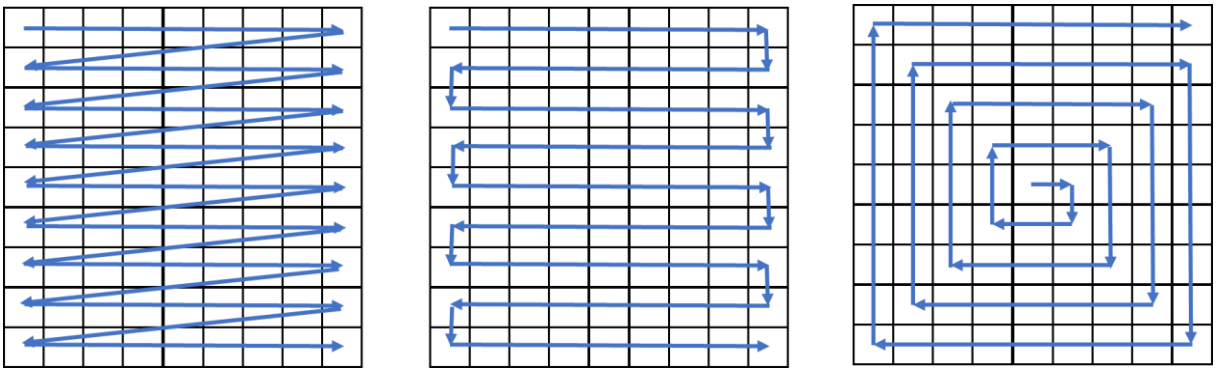


Figure 3.14. Examples of scanning orders: Raster (left), Serpentine (center), Spiral (right).

3.4.2. Multiview-based approaches

Several multiview-based approaches have been proposed in the state-of-the-art where all the SAIs can be interpreted as a HDCA signal and encoded with a multiview coding solution. The main advantage of this solution is that it is compatible with HDCA signals allowing for a single solution for encoding both types of LFs. Consequently, coding standards like MVC or MV-HEVC can be directly applied to these signals, allowing for high coding efficiency. These codecs allowed for intra-SAI redundancy to be exploited by the standard intra prediction modes in both H.264 and HEVC, as well as inter-SAI redundancy to be exploited by inter-view, i.e., disparity prediction modes [63]. Finally, in case LF video is being encoded, temporal redundancy is exploited using motion prediction modes.

Two types of configurations are typically used: i) each SAI is considered to be an individual view [69]–[72], allowing LF video to be encoded as well; ii) The LF image is organized in multiple PVS with an equal amount of frames, where each PVS is considered to be a different view [73], [74], as illustrated in Figure 3.15. In both types of configuration it is possible to have a 2D prediction structure relating several SAIs. However, in the case of the second configuration, part of the inter-SAI prediction is applied on the time domain. Some of the

proposed approaches to encode LF images based on the multiview coding standards [73], [74], also include improved 2D rate allocation schemes. Alternatively, in [75], the second configuration type is used, however, the authors use a neural network to synthetize virtual references from reconstructed neighbor frames. The previously encoded SAIs in both horizontal and vertical directions, relative to the current SAI are used to feed the neural network, which then generates the desired virtual reference images to improve the encoding process. The 2D prediction structure was modified in order to maximize the provided number of virtual references to each SAI that is being encoded.

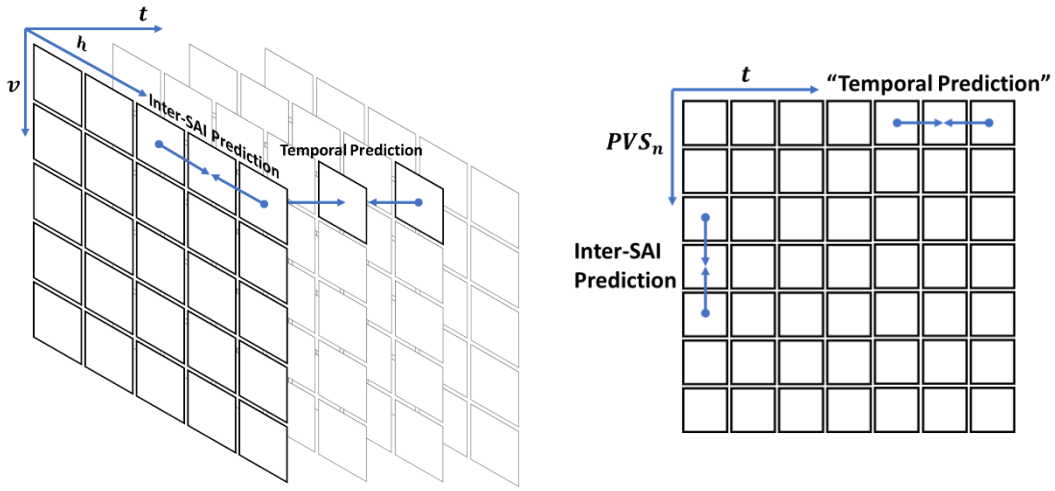


Figure 3.15. Two possible configurations to encode LF image and video that use multiview coding approaches where each SAI is considered an individual view (left) and multiple PVS are encoded as a multiview video (right).

3.4.3. SKV-based approaches

When exploiting inter-SAI redundancy, the main factor that contributes to higher redundancy is the disparity between each SAI. In scenarios where the LF is captured by a hand-held LF camera, i.e., using an MLA, the baseline between the several microlenses is very small, which typically contributes to very low disparities between each SAI. Consequently, several authors have proposed to only encode and transmit part of the SAIs, normally referred to as SKVs, and then transmitting additional information in the bitstream to the decoder to generate the remaining non-SKVs. These approaches are normally structurally similar, as shown in Figure 3.16.

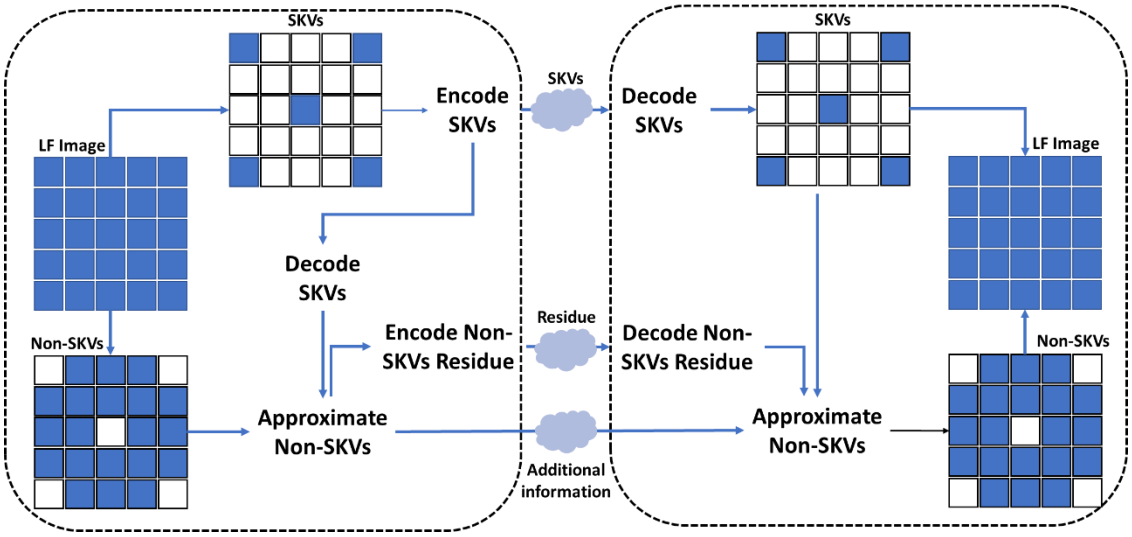


Figure 3.16. SKV-based LF image coding approach structure.

In Figure 3.16, the SAIs that compose the LF image are split into two groups, SKVs and non-SKVs. The SKVs are encoded directly and an approximation of the non-SKVs is generated, based on the decoded SKVs and some additional information extracted from the original non-SKVs. The residual image of the non-SKVs is also encoded and transmitted. In order to decode the non-SKVs, it is only necessary to transmit the SKVs, the residual image of the non-SKV and some additional information gathered by the technique used to approximate the non-SKVs. The differentiating factor between the several SKV-based approaches is the distribution of SKVs and non-SKVs and the technique used to approximate the non-SKVs. The following non-SKVs approximation approaches can be found in the literature:

- **Linear approximation:** In [76]**,** the SKVs and non-SKVs are first separated in a checkboard pattern, i.e., the SKVs are about half the total number of SAIs. Firstly, a view approximation model is constructed assuming that the disparity between SAIs is linearly distributed, i.e., the disparity is the same for two pairs of SAIs with the same baseline between them. Secondly, once the disparity between SAIs is known, the SKVs are warped to the current non-SKV position which are then combined using a weighted sum, generating an approximation of the non-SKV.
- **Depth/disparity image based rendering:** In [77] the disparity maps of the four corner SAIs (SKVs) are estimated using optical flow. The remaining disparity maps (the disparity maps of the non-SKVs) are generated by forward warping of the four corner

SKVs disparity maps. DIBR is then applied to synthesize the non-SKVs. Alternatively, in [78] five SKVs are selected from the LF image, which correspond to the central SAI, and border SAIs in the upper, lower, left and right position relative to the central SAI. The LF is approximated with disparity guided sparse coding over a perspective shifted LF dictionary based on the SKVs. The non-SKVs are generated using approximated disparity maps, while the SKVs and residuals of the non-SKVs are transmitted to the decoder. Finally, in [79], the SKVs are selected as seven sequences of SAIs (PVS) that correspond to the odd columns of SAIs present in a 13×13 LF image. The SKVs are encoded using MV-HEVC, similarly to the second configuration mentioned in the multiview-based approaches in the previous section. The estimated depth maps of the SKVs are then used on the decoder to generate the approximated non-SKVs.

- **Neural network-base prediction:** In [80], four SKVs are selected from the LF image to be encoded as a PVS. The decoded SKVs feed a CNN which applies angular super resolution to generate predictions of the non-SKVs. The residuals between the predicted non-SKVs and the original non-SKVs are encoded and added to the bitstream. The same CNN is used at the decoder to generate the same predicted non-SKVs, based on the four decoded SKVs. In [81], the SKVs are selected as a checkered pattern and encoded as a PVS. The non-SKVs are predicted using a generative adversarial network (GAN). The residuals between the original non-SKVs and the predictions provided by the GAN are transmitted to the decoder with the compressed SKVs.

- **Graph-based prediction:** In [82], the four corner SAIs are selected as SKVs as in [77]. The SKVs are fed to a CNN in order to generate predictions of the remaining non-SKVs. The residuals between such predictions and the original non-SKVs are then grouped into super pixels, i.e., group of connected pixels with similar colors, using the super linear iterative clustering (SLIC) algorithm [83]. Afterwards, graph transforms are applied on each super-pixel followed by quantization and entropy coding, generating the bitstream. Alternatively, in [84], a graph is estimated between all the SAIs that compose the LF image. The encoder selects a subset of SAIs (about half of the SAIs, in a checkerboard pattern) that are compressed (SKVs) using a lossy view encoder, while the graph weights are encoded using a lossless graph encoder. The non-SKVs are generated in the decoder using the information from both the decoded SKVs and the decoded graph weights.

3.4.4. Transform-based approaches

As in MI-based methods, transforms, such as a three-dimensional discrete wavelet transform (3D-DWT), can also be used to exploit the LF redundancy [85]. In this case, the LF image is decomposed into SAIs, and a 3D-DWT is applied to the stack of SAIs. The lower frequency bands are transformed using a two-dimensional discrete wavelet transform (2D-DWT), while the remaining higher frequency coefficients are simply quantized and arithmetically encoded. The 2D-DWT lower frequency bands of several SAIs are grouped into a 3D block, as shown in Figure 3.17. A 3D-DCT is applied to the generated 3D block and the DCT coefficients are quantized and encoded using Huffman coding.

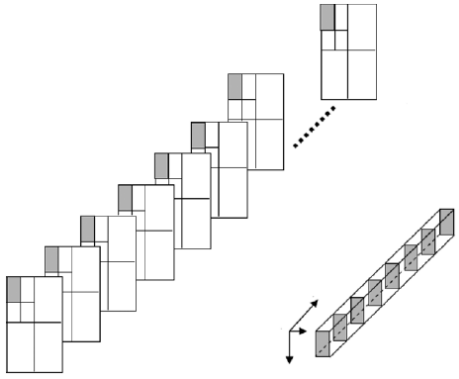


Figure 3.17. 2D-DWT lower bands grouping example: 8 SAIs are grouped generating an 8×8×8 block [85].

3.5. Alternative-representation-based approaches

Alternatively to the MI- and SAI redundancy based techniques, some approaches are based on different data representations, allowing different types of redundancy to be exploited. Some examples of such representations include:

- **Homography-based low-rank approximation:** In homography-based low-rank approximation (HLRA) approaches [78, 79], there is a joint search for a set of homographies that best align the SAIs. The SAIs can be aligned using either one global homography or multiple homographies, depending on how much the disparity across SAIs varies from one depth plane to the other. All the warped SAIs using the selected homographies are then vectorized and concatenated in a $M \times N$ pixels matrix, where N is the number of SAIs and M is the number of pixels per warped SAI. A joint optimization of homography parameters and low rank matrix approximation is

performed for a target K rank. The low rank approximation is represented as $B \times C$, where B has a size of $M \times K$ and C has a size of $K \times N$, which is smaller (lower rank) than the initial representation $M \times N$ if $K < N$. Each column of matrix B is encoded using HEVC-Intra while the weights contained in C and the homography parameters are encoded using scalar quantization followed by Huffman coding.

- **Super-Ray based low rank approximation:** In [88] a super-ray low rank approximation is used to improve LF image coding. Firstly, a SLIC segmentation technique is used to create the SAI super-pixels. Secondly, using disparity information, the super-rays are constructed by connecting the several super-pixels across the several SAIs using the central SAI as the reference. Thirdly, disparity compensation is applied to each super-ray, in order to align them across the multiple SAIs. Finally, a low rank approximation is computed using singular value decomposition (SVD) applied to the aligned super-ray representation resulting in two sets of eigen images, corresponding to the light rays visible in the central view and the occlusions. This information is transmitted to the decoder together with a matrix of coefficients and side information that allow the inverse alignments of super-rays in the decoder.

3.6. Final remarks

This chapter briefly reviews the HEVC video coding standard and JPEG Pleno Part 2, as well as the most relevant LF image coding techniques available in the literature. HEVC is presented because, although it is not generally the most efficient solution to encode LF images, it is the base of several state-of-the-art LF image coding techniques that rely on a MI- or SAI-based representation.

JPEG Pleno Part 2 introduces two different approaches to encode LF images: MuLE that is based on 4D Transform mode and WaSP that is based on 4D prediction. MuLE is typically more efficient for MLA-based LF images and WaSP is more efficient for LF images captured using HDCAs.

Both MI and SAI representations allow to exploit different types of redundancy, which can be achieved by using different image coding techniques. The available techniques in the literature can be organized by the type of representation:

- **MI-based approaches** are able to more efficiently exploit the intra- and inter-MI redundancy which is especially relevant for MLA-based LF images. Most solutions are able to encode the LF image without prior knowledge about the type of capturing device. The presented approaches can be categorized based on the type of technique used to exploit intra-MI and inter-MI redundancy, namely: search algorithms, spatial transforms, MI reshaping and disparity compensation.
- **SAI-based approaches** require a prior knowledge about the type of capturing device in order to correctly convert the MI representation into a SAI representation. However, once the LF image is converted to SAIs, several approaches based on existent video and multiview coding standards can be used to efficiently encode the LF image. Such approaches are able to exploit the intra-SAI and inter-SAI redundancy being categorized based on the type of technique used to exploit these types of redundancy: PVS, multiview, SKV and spatial transforms.

Finally, some techniques use alternative representations and, therefore, different types of redundancy that cannot be categorized as solely MI- or SAI-based, namely low rank approximations based on homographies applied to SAIs and super-rays.

Chapter 4. Light field representation comparison

State-of-the-art LF image coding solutions, usually, rely in one of two LF data representation formats: LL or 4D LF. While the LL data representation is a more compact version of the LF, it requires additional camera metadata and processing steps prior to image rendering. On the contrary, 4D LF data, consisting of a stack of SAI, provides a more redundant representation requiring, however, minimal side information, thus facilitating image rendering.

As mentioned in Chapter 2, JPEG Pleno CTCs, regarding the objective evaluation of LF image coding defined a processing chain that allows to compare different 4D LF data codecs, aiming to facilitate codec assessment and benchmark. Thus, any codec that does not rely on the 4D LF representation needs to undergo additional processing steps to generate an output comparable to a reference 4D LF image. These additional processing steps may have impact on the quality of the reconstructed LF image, especially if color subsampling format and bit depth conversions have been performed. Consequently, the influence of these conversions needs to be carefully assessed as it may have a significant impact on a comparison between different LF codecs.

The remainder of this chapter is organized as follows: Section 4.1 presents an exhaustive comparative analysis of the processing chains of LL and 4D LF data representation formats, considering different color subsampling formats and bit depths; Section 4.2 shows how each processing chain affects the coding efficiency; Finally, Section 4.3 concludes this chapter with some final remarks.

4.1. Processing chain for objective quality assessment

JPEG Pleno CTCs proposes a reference processing chain to assess the coding efficiency of different LF coding solutions (see Figure 4.1). This processing chain aims to accommodate

different LF coding solutions that may rely on different LF data representations [35]. However, objective and subjective performance assessment is done using a common data representation: the 4D LF data representation [35]. Therefore, regardless of the coding approach, the corresponding output format must be converted into a 4D LF, organized as a stack of RGB 4:4:4 10 bit/sample SAIs, i.e., the same format of the reference 4D LF image generated using the reference processing chain shown in Figure 4.1.

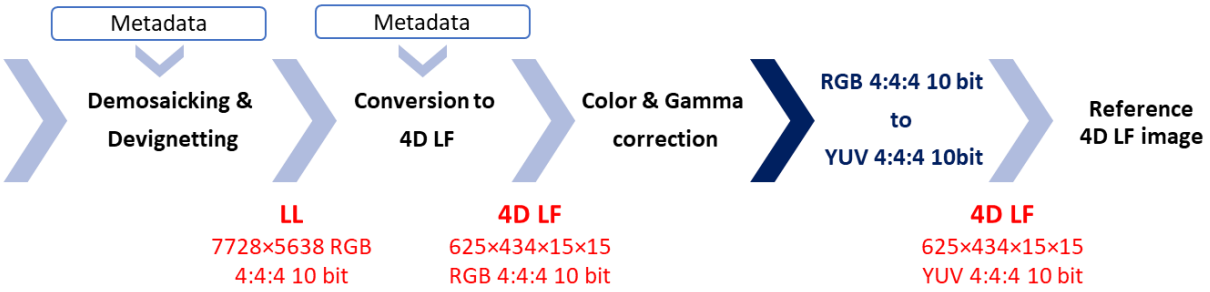


Figure 4.1. JPEG Pleno reference LF processing chain.

The JPEG Pleno reference LF processing chain is comprised by the steps described in Chapter 2 to generate the LL and 4D LF data representation from the raw LF data. To assess the objective quality of a LF image codec, it is required to encode and decode the LF image using a coding solution and then converting the LF image to the 4D LF data representation using a RGB 4:4:4 10 bit/sample color format. If the codec is limited to a specific LF data representation, e.g., LL, and or a specific color format, e.g., YUV 4:2:0 8 bit/sample, the processing chain needs to be adapted.

The following sections will tackle the processing chain when two data representations are tested: LL and the 4D LF common data representation. Additionally, for both data representations two color subsampling formats with different bit depths are also compared, YUV 4:4:4 10 bit/sample (10 bit) and YUV 4:2:0 8 bit/sample (8 bit).

4.1.1. Processing chain for the LL data representation

When the LL data representation is used in a given codec, since it is different from the 4D LF common data representation, some adaptations to the processing chain are necessary. These adaptations will allow the output LF image to use the same data representation and color format as the reference 4D LF image, and therefore to be comparable in terms of objective metrics.

Additionally, since each codec might be designed for a specific color format, the required color conversions for the specific color format must also be included.

The processing chain for a LL data representation with an YUV 4:4:4 10 bit format is shown in Figure 4.2.

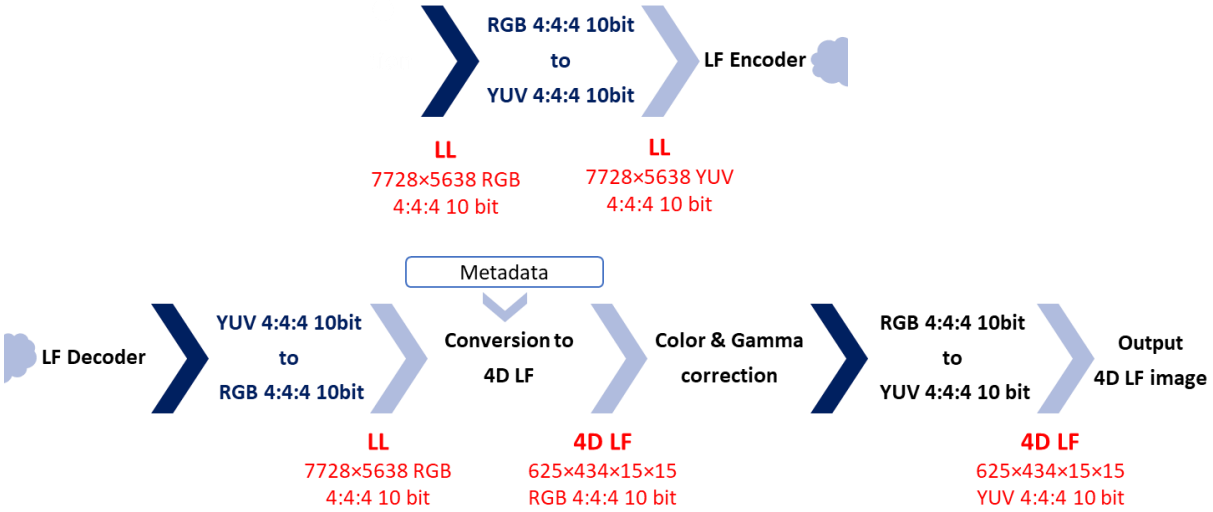


Figure 4.2. Processing chain to encode and decode LF data with LL data representation and YUV 4:4:4 10 bit format.

In the case of Figure 4.2, the codec uses the YUV 4:4:4 10 bit format, therefore the necessary color conversion is applied prior to the encoding step. After encoding and decoding the LL LF image, a conversion to the 4D LF data representation is necessary. The LF Toolbox [34] for such conversion requires the input to be in RGB 4:4:4 10 bit format, therefore the decoded LL LF image is converted to such specific color format and then converted to 4D LF data representation. Color and Gamma correction is applied to the 4D LF image that is, finally, converted to YUV 4:4:4 10 bit in order to be compared to the reference 4D LF image in terms of objective metrics.

When a different color format is required, e.g., YUV 4:2:0 8 bit, some changes must be done in the processing chain, as shown by the yellow arrows in Figure 4.3.

In Figure 4.3, as the encoded LL LF image is in the YUV 4:2:0 8 bit format, the image color quality may be potentially degraded, but the amount of information to be compressed is greatly reduced.

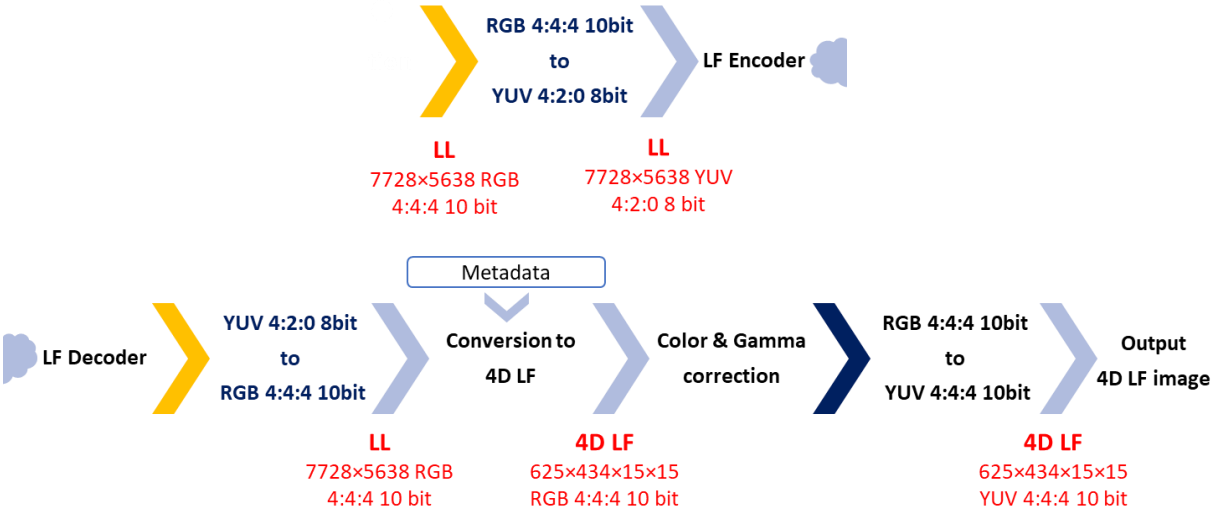


Figure 4.3. Processing chain to encode and decode LF data using the LL data representation and YUV 4:2:0 8 bit color format.

When using the LL data representation, some additional information, i.e., camera metadata [89], has to be transmitted to the decoder, in order to allow a proper view rendering. Converting the LL data representation into 4D LF facilitates the view rendering [89]. This metadata must include, at least, the MI center coordinates and MI size [14]. Any alternative to the 4D LF data representation format requires a processing step to convert the specific data representation to 4D LF using RGB 4:4:4 10 bit, in order to be compared with the reference 4D LF image.

4.1.2. Processing chain for the 4D LF data representation

To encode the LF image using a variation of the 4D LF data representation, enables to use a simpler processing chain than that used to encode the LL data representation. In Figure 4.4 and Figure 4.5, the required processing chain for 4D LF is shown for the encoder and decoder, respectively.

In this study only two variants of the 4D LF are considered: 4DLF-MI and 4DLF-PVS. The 4DLF-MI consists on a 2D frame with all MIs concatenated as a matrix and the 4DLF-PVS is a sequence of SAIs (pseudo-video) using a spiral scan [64]. Although the formats are different, the conversion between both is seamless, without losing any information. However, to encode the LF image using the 4DLF-PVS variant by, efficiently exploiting the redundancy between each SAI, a video codec is necessary.

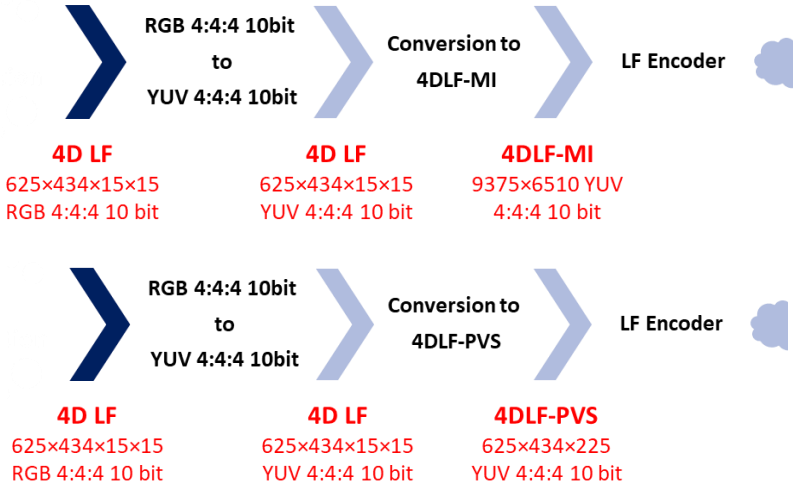


Figure 4.4. Processing chain to encode LF data using the 4DLF-MI (top) and 4DLF-PVS (bottom) data representation and YUV 4:4:4 10 bit color format.

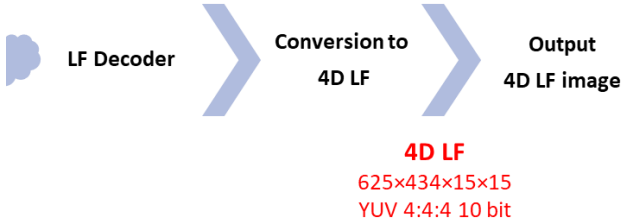


Figure 4.5. Processing chain to decode LF data using the 4DLF-MI and 4DLF-PVS data representation and YUV 4:4:4 10 bit color format.

In the case of Figure 4.4, the 4D LF image is converted into the color format required by the codec, in this case YUV 4:4:4 10 bit. After this step, the image is converted into the 4D LF variant 4DLF-MI or 4DLF-PVS. Finally, the resulting LF data is encoded, decoded, converted to 4D LF and compared to the reference 4D LF image.

When the codec is limited to the color format YUV 4:2:0 8 bit, the arrows in yellow in Figure 4.6 and Figure 4.7 show the added block to the processing chain.

In this case, since the required color format is YUV 4:2:0 8 bit, a color conversion block is added to the processing chain, allowing the output LF image to be compared with the reference 4D LF image.

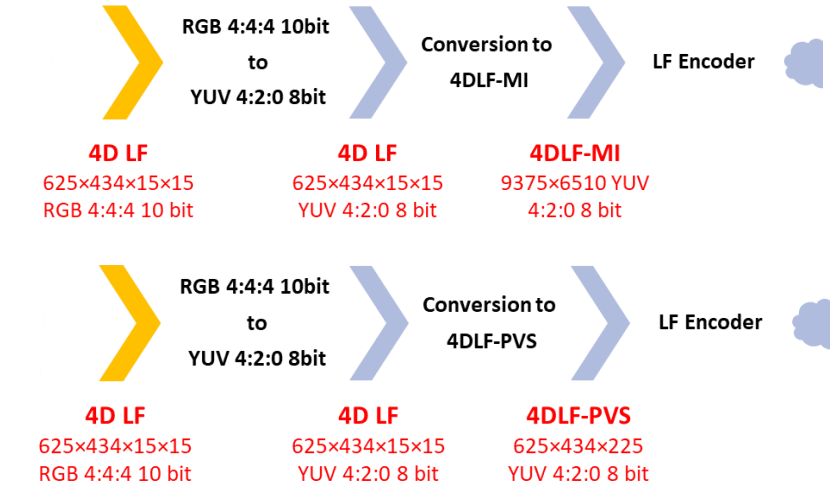


Figure 4.6. Processing chain to encode LF data using the 4DLF-MI (top) and 4DLF-PVS (bottom) data representation and YUV 4:2:0 8 bit color format.

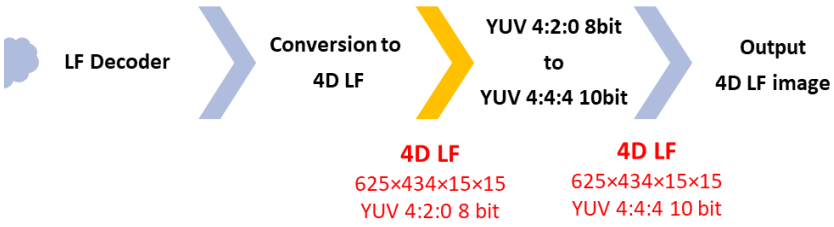


Figure 4.7. Processing chain to decode LF data using the 4DLF-MI and 4DLF-PVS data representation and YUV 4:2:0 8 bit color format.

4.1.3. Comparison of LL and 4D LF data representations

The following major differences between the LL and 4D LF data representations may drastically impact on the objective performance assessment:

1. **Camera metadata**: By effectively converting the LF data into a stack of SAIs, the 4D LF data representation does not require any camera metadata to perform basic rendering tasks at the decoder side, namely, view generation and focus change. However, when using the LL data representation, some specific camera metadata is required to enable proper rendering tasks. This information is not standardized; however, it could include transmitting the MI size and MI centers, which are necessary to convert from the non-integer hexagonal grid to the integer square grid of MIs.

2. **LF image resolution**: In the case of the LL data representation, the LF image resolution will be the native resolution, i.e., 7728×5368 pixels for the Lytro Illum. However, when this format is converted to the 4D LF data representation, the number of pixels is

expanded to 9375×6510 pixels, or 625×434×15×15 pixels, when using 4DLF-MI or 4DLF-PVS data representations, respectively, which represents an increase of around 47% in the amount of data to be transmitted to the decoder. Therefore, the LL data representation is much more compact if camera metadata is not considered.

3. **Color processing**: Due to the color conversion and color and gamma correction processing steps, which are necessary for both 4D LF and LL data representations, color degradation along the processing chain is likely to occur. In such case, the LL data representation is likely to suffer from a more prevalent degradation of color components when compared to the 4D LF data representation, due to two main reasons: i) The higher number of color conversions steps for both YUV 4:4:4 10 bit and YUV 4:2:0 8 bit; ii) the color and gamma correction step is only performed on the decoder side after the losses introduced by the encoder and by the color conversion processing steps, reducing the final color accuracy.

4.2. Experimental results

In this section, the LF data representations described in the previous sections, namely LL, 4DLF-MI and 4DLF-PVS, are used. Additionally, for each LF data representation two color formats, YUV 4:4:4 10 bit and YUV 4:2:0 8 bit, are used in the codec block of the processing chain. With these combinations of LF data representation and color formats, the effects of the suggested processing chain are evaluated in terms of objective results. The codec used in these experimental tests is the HM-16.9 implementation of HEVC-RExt [90], that may use different coding profiles, depending on the combination of the LF data representation and color format. The following six scenarios were tested, along the specific coding profiles and used processing chains:

1. **LL YUV 4:2:0 8 bit**, using Main Intra profile and the processing chain of Figure 4.3;
2. **4DLF-MI YUV 4:2:0 8 bit**, using Main Intra profile and the processing chain of Figure 4.6;
3. **4DLF-PVS YUV 4:2:0 8 bit**, using Main profile and the processing chain of Figure 4.6;
4. **LL YUV 4:4:4 10 bit**, using Main 4:4:4 10 Intra profile and the processing chain of Figure 4.2;

5. **4DLF-MI YUV 4:4:4 10 bit**, using Main 4:4:4 10 Intra profile and the processing chain of Figure 4.4;
6. **4DLF-PVS YUV 4:4:4 10 bit**, using Main 4:4:4 10 profile and the processing chain of Figure 4.4.

In the following sections, the specific block settings for each processing chain are described, as well as the achieved experimental results and individual conclusions.

4.2.1. Color correction

The reference LF images were generated by the reference processing chain, shown in Figure 4.1, using the LF Toolbox v0.4 [34] for the Demosaicking, Devignetting and Conversion to 4D LF data representation blocks. For the Color and Gamma correction blocks, the configuration suggested in JPEG Pleno CTCs [35] was used.

$$ColorMatrix = \begin{bmatrix} 2.2172 & -0.4233 & -0.1989 \\ -1.0467 & 1.7511 & -0.7366 \\ -0.1706 & -0.3278 & 1.9354 \end{bmatrix}; \quad (4.1)$$

$$ColorBalance = [1\ 1\ 1]; \quad Gamma = {}^{1}/_{2.2}$$

After applying the Color and Gamma correction defined by (4.1), the reference 4D LF image was generated, in order to be compared with the output LF images, either from the LL or 4D LF processing chains.

4.2.2. Chroma and bit depth upsampling and downsampling

Regardless of using the LL or 4D LF data representations, the color conversion blocks are necessary to convert the LF images to YUV 4:4:4 10 bit or YUV 4:2:0 8 bit. Additionally, the conversion from RGB to YUV is done using the Recommendation ITU-R BT.709-6, as it was described in the JPEG Pleno CTCs [35].

Additionally, subsampling the chroma components from YUV 4:4:4 to YUV 4:2:0 and upsampling from YUV 4:2:0 to YUV 4:4:4 is done using scripts made available for the ICME 2016 Light Field Challenge [91]. Bit depth downsampling from 10 bit to 8 bit is performed by truncating the two least significant bits. Bit depth upsampling from 8 bit to 10 bit is done by applying a left bit shift of 2 bits.

4.2.3. Objective performance assessment

In order to access the objective performance of the codec, in this case HEVC-RExt, for six different scenarios, it is necessary to: i) calculate the number of bits per pixel the codec uses to encode the LF image plus the metadata necessary to perform image rendering; ii) calculate the average PSNR of each rendered SAI, in comparison to the SAIs rendered from the reference 4D LF.

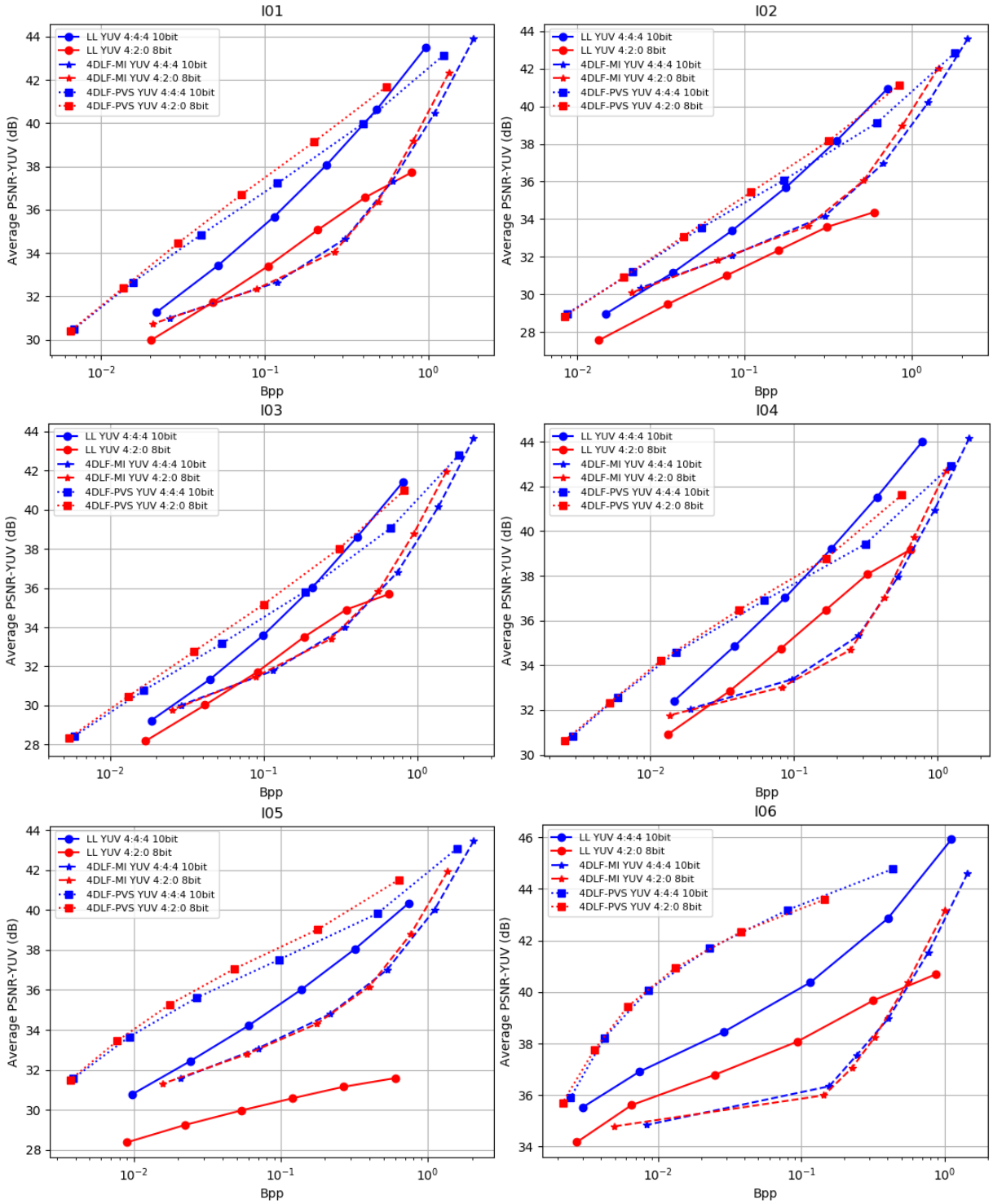
Since the EPLF dataset [22] is based on LF images, captured using the Lytro Illum LF camera, the number of possible SAIs that can be extracted using the LF Toolbox is 15×15. However, since the outer views are mostly unusable, JPEG Pleno recommends that only the results for the inner 13×13 views are evaluated [35]. Because of this, when the 4D LF data representation is used, the outer views are discarded before the encoding step, so effectively, only 13×13 views are encoded. Each view has a resolution of 625×434 pixels, thus the total number of pixels is 45841250. Although there is a significant reduction in the number of views (approximately 25%), i.e., from 225 (15×15) views to 169 (13×13) views, the spatial resolution used to encode the 4D LF data representation is still, approximately 11% larger than the one used by the LL data representation. The number of bits per pixel is calculated by dividing the number of bits used to encode the LF image by the total number of pixels, i.e., 45841250.

As it was explained in the previous section, regardless of the used data representation, the LF images that reach the decoder side are converted into 4D LF, with the color format YUV 4:4:4 10 bit, to be compared to the reference 4D LF image. The objective quality is obtained by calculating the individual PSNR-YUV for each SAI, comparing the output and the reference 4D LF image, and then averaging the 13×13 individual PSNR-YUV values as described in Section 2.6.2.

Every image from the EPFL LF dataset (see Appendix A.2) was encoded and decoded using HEVC-RExt, with the appropriate profile. The QPs used for the LL and 4DLF-MI data representations were 22, 27, 32, 37, 42 and 47, and the QPs used for 4DLF-PVS were 17, 22, 27, 32, 37 and 42. The achieved results for this selection of QPs, for both representations, roughly produce bitrates within the target bit rates proposed by JPEG Pleno CTCs document, i.e., between 0.001 bpp and 0.75 bpp [35] .

4.2.4. Analysis of the experimental results

The experimental results for the full EPFL LF dataset using HEVC-RExt codec, in the six scenarios previously listed are shown in Figure 4.8.



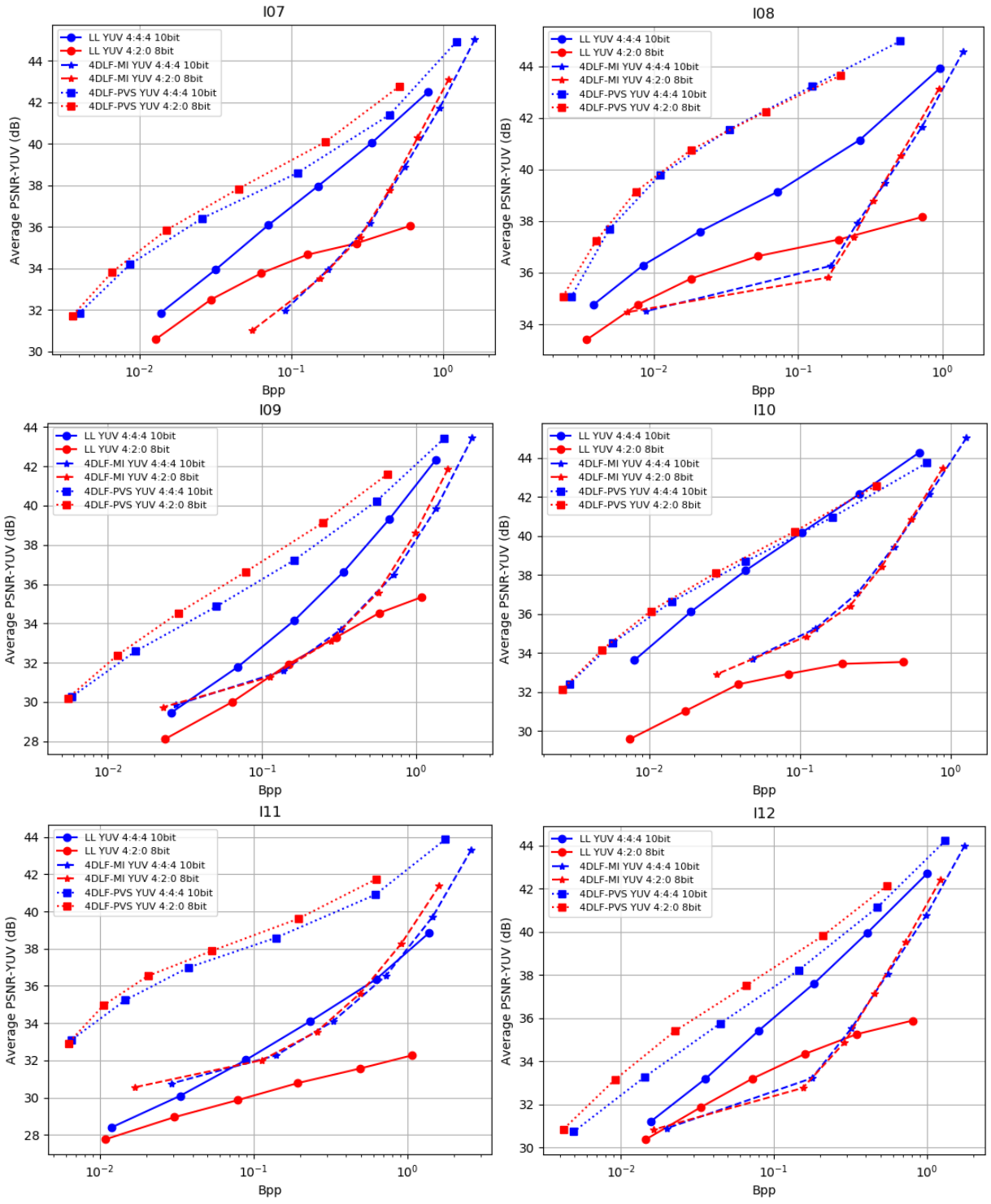


Figure 4.8. Rate-Distortion results comparing all six combinations of LF data representations and color formats for the twelve LF images from the EPFL LF dataset.

LF data representation comparison

When comparing the results shown in Figure 4.8 in terms of LF data representation, it is possible to observe that the least efficient LF data representation is the 4DLF-MI. In general, the 4DLF-MI, regardless of the color format, and LL, using YUV 4:2:0 8 bit color format, achieve the worst performance in comparison to the remaining LF data representations. It indicates that the redundancy that exists within neighboring MIs is not properly exploited by HEVC. Both data representations are expected to perform significantly better if using a prediction tool to exploit the MI redundancy in HEVC [51].

The LL format, using YUV 4:4:4 10 bit and 4DLF-PVS, regardless of the color format, provide higher efficiency, but in general the 4DLF-PVS allows for better results. It is possible to observe that the LL data representation tends to be more competitive against the 4DLF-PVS at higher bitrates, namely, when encoding images I01, I02, I03, I04 and I10, It is worth reinforcing the fact that no metadata is being considered for the LL data representation, and that the 4D LF based data representations generate roughly 11% more pixels to encode. Moreover, when using HEVC to encode the 4DLF-PVS, the intra- and inter-SAI redundancy is exploited, which does not happen when HEVC encodes LL or 4DLF-MI data representations due to the lack of adequate coding tools.

Color format comparison

When strictly comparing color formats within each LF data representation, there is a clear tendency of the 4D LF to be more efficient to encode YUV 4:2:0 8 bit color when compared to the YUV 4:4:4 10 bit. This can be seen in both 4DLF-PVS data representation, for all images, and for 4DLF-MI data, for images I02, I03, I07, I09 and I11, while for the remaining images both color formats achieve similar coding efficiencies.

The exact opposite tendency can be seen in the case of the LL data representation, where using YUV 4:4:4 10 bit is more efficient than using YUV 4:2:0 8 bit. This can be seen for all images, across every tested bitrate.

These results can be justified by the use of color correction and color conversions processing block. In the case of 4D LF data representation, the color correction processing step is applied early in the processing chain, before any color conversion or encoding takes place. As the color conversion and encoding steps add distortion to the LF image luma and chroma components,

and since the color correction was applied before chroma subsampling and encoding, in the case of YUV 4:2:0 8 bit, the objective quality of these components is higher.

In the case of the LL data representation, more color conversion steps are necessary, especially when YUV 4:2:0 8 bit color format is used. As the color correction step, for the LL format, is only applied on the decoder side, the chroma components carry the distortion introduced by color conversion and encoding steps.

Maximum objective quality

In order to encode 4D LF RGB 4:4:4 10 bit, it is necessary to add processing blocks that introduce irreversible distortion to the LF image, regardless of the coding method. In order to assess the maximum objective quality, the processing chains used in the last section for the six possible scenarios are reused but removing the codec processing block. The maximum objective quality for both LL and 4D LF data representations, using YUV 4:4:4 10 bit and YUV 4:2:0 8 bit color formats, are shown in Table 4.1 and Table 4.2 for the average PSNR-Y and average PSNR-YUV, respectively.

Table 4.1. Maximum objective quality measured in Average PSNR-Y (bold and italic PSNR values correspond to the maximum and minimum, respectively).

Avg. PSNR-Y (dB)	**I01**	**I02**	**I03**	**I04**	**I05**	**I06**	**I07**	**I08**	**I09**	**I10**	**I11**	**I12**	**AVG.**
LL YUV 4:2:0 8 bit	44.55	39.17	41.92	**48.06**	*35.68*	47.65	40.07	41.31	41.10	37.23	35.96	38.77	40.96
4D LF YUV 4:2:0 8 bit	58.49	*58.45*	58.46	58.45	58.57	58.45	**60.01**	58.45	58.48	58.51	58.45	58.45	58.60
LL YUV 4:4:4 10 bit	65.23	61.47	62.36	65.51	59.42	**66.21**	59.29	59.85	63.93	62.11	*56.10*	60.39	61.82
4D LF YUV 4:4:4 10 bit	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.

Table 4.2. Maximum objective quality measured in Average PSNR-YUV (bold and italic PSNR values correspond to the maximum and minimum, respectively).

Avg. PSNR-YUV (dB)	**I01**	**I02**	**I03**	**I04**	**I05**	**I06**	**I07**	**I08**	**I09**	**I10**	**I11**	**I12**	**AVG.**
LL YUV 4:2:0 8 bit	41.86	36.97	39.09	44.41	34.33	**45.08**	38.41	40.06	38.95	35.82	*34.29*	37.03	38.86
4D LF YUV 4:2:0 8 bit	54.52	54.24	54.01	54.75	54.50	54.95	**55.16**	54.82	54.23	54.94	*53.46*	54.38	54.50
LL YUV 4:4:4 10 bit	62.34	58.48	59.16	62.38	57.12	**63.87**	57.63	58.67	61.04	55.66	*54.16*	58.12	59.05
4D LF YUV 4:4:4 10 bit	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.

From Table 4.1 and Table 4.2 it is possible to conclude that the processing chain for the 4D LF data representation has a smaller impact on the final average PSNR. This is justified by the fact that the alternative LF data representations, in this case the LL, need additional processing steps that introduce distortion to the LF image.

The average PSNR-Y is higher than the average PSNR-YUV for every tested case, which is justified by the additional color conversions that are applied to each individual processing chain. Although this is true for both LL and 4D LF data representations, in the case of the LL data representation, the average PSNR-YUV reaches values as low as 34 dB when YUV 4:2:0 8 bit color format is used and 54 dB when YUV 4:4:4 10 bit color format is used.

The average PSNR-YUV for all images is roughly 15 dB higher for the 4D LF data representation in comparison with the LL data representation. This confirms that the LL data representations is more affected by the processing chain than the 4D LF data representation, in terms of both luma and chroma components.

In this case, no distinction is done between 4DLF-MI and 4DLF-PVS, because the conversion between both of these 4D LF data representation is reversible and can be applied at any point in the processing chain.

In the case of the 4D LF data representation used with YUV 4:4:4 10 bit color format, when removing the codec processing block, the processing chain becomes identical to the reference processing chain, consequently, the PSNR is infinite for every case.

Summary of the experimental results

The upper triangle in Table 4.3 shows the comparisons between the tested LF data representations in terms of color degradation and maximum objective quality. While, the lower triangle compares them in terms of achieved coding efficiency for each color format.

4.3. Final remarks

This chapter presented an in-depth comparison of the effects of using existing LF data representations, namely, LL, 4DLF-MI and 4DLF-PVS, considering different color formats

Table 4.3. Summary of the LF data representation comparisons

VS	4DLF-MI	4DLF-PVS	LL
4DLF-MI		Both LF data representations achieve the same **color degradation** and **maximum objective quality**	4DLF-MI has less **color degradation** and achieves a higher **maximum objective quality**
4DLF-PVS	4DLF-PVS is more efficient for every image that was tested regardless of the color format		4DLF-PVS has less **color degradation** and achieves a higher **maximum objective quality**
LL	**YUV 4:4:4 10 bit**: LL is more efficient for every image that was tested **YUV 4:2:0 8 bit**: LL is in general more efficient for low bitrates	**YUV 4:4:4 10 bit**: 4DLF-PVS is more efficient, with the exception of I01, I02, I03, I04, 10 for higher bitrates **YUV 4:2:0 8 bit**: 4DLF-PVS is more efficient for every case	

namely YUV 4:4:4 10-bit and YUV 4:2:0 8-bit, using JPEG Pleno CTCs. Six individual combinations where tested each one requiring a specific encoding and decoding processing chain, and finally converted to the common format for objective quality assessment, i.e., 4D LF YUV 4:4:4 10-bit [35].

The experimental results that have been performed using the HEVC-RExt codec [90], show that the 4DLF-MI data representation achieves the lowest coding efficiency, regardless of the color format. Although the same number of pixels are being encoded, the 4DLF-PVS achieves a higher coding efficiency, when compared to 4DLF-MI, by exploiting the inter-SAI redundancy using the temporal predictions tools available in HEVC. The 4DLF-PVS is also more efficient, in general, than the LL data representation. This can be observed for every image and bitrate tested, with the exception of the higher bitrates for five out of the twelve tested images. In such cases, the LL data representation is only able to outperform the 4DLF-PVS data representation when using the YUV 4:4:4 10 bit color format.

The LL data representation benefits from using the color format YUV 4:4:4 10-bit while both 4D LF data representations produce better results when YUV 4:2:0 8-bit color format is used. This conclusion can be further verified by comparing the results using the average PSNR-Y and average PSNR-YUV. When analyzing the results for average PSNR-Y, the efficiency gains of the YUV 4:2:0 8-bit color format are even more notorious than when using the average PSNR-YUV. However, when performing the same analysis using the 4D LF data representation, the opposite tendency occurs; the efficiency gains of the YUV 4:4:4 10-bit are lower when using the average PSNR-Y, instead of average PSNR-YUV.

When analyzing the results for the maximum quality of each of the six tested scenarios, with the different LF data representations and color formats, one can conclude that the maximum quality is lower when the LL data representations is used. When the YUV 4:2:0 8-bit color format is used, the maximum quality using the average PSNR-YUV can be as a low as 34 dB. When analyzing the maximum quality results using the average PNSR-Y metric, the objective quality is always higher, with an average increase of about 3 dB. These results lead to the conclusion that, an alternative LF data representation like LL, although more compact than the 4D LF variants, suffers more color degradation due to the more extensive processing chain.

Chapter 5. Light field image coding using high order intra block prediction

It was seen in the LF representation study presented in Chapter 4, that the MI-based representations tends to be less efficient than the SAI-based ones when being directly encoded by HEVC. Therefore, developing coding techniques that will exploit the MI redundancy which is exclusive to LF content, is likely to produce high gains in coding efficiency because it is not being exploited.

As it was shown in Chapter 3, several state-of-the-art LF coding schemes rely on search algorithms mostly based on block matching or template matching techniques to exploit the inherent intra- and inter-MI redundancy in LF images. Such search algorithms can be unidirectional, bidirectional, based on locally linear embedding or gaussian process regression. However, the common denominator in these approaches is that all these coding schemes use low order prediction (LOP) which only allow two DoF, as only translations are used to describe the inherent LF image inter-MI redundancy.

Due to the small baseline between MIs in lenslet LF images, the different MIs can be approximately related by changes in perspective, which require eight DoF to be described. This additional matching accuracy is important to develop a coding method able to cope with important features of the LF content, such as:

- The LF camera model, i.e., both focused and unfocused models should be handled;
- The type of microlens array structure, e.g., rectangular or hexagonal microlens layouts, creating rectangular, hexagonal or circular MIs;

- The MI size, i.e., a parameter that depends on the camera, and has a strong influence on the number of possible rendered SAIs and their spatial resolution.
- The LF representation, i.e., any MI based data representation, (e.g., LL or 4DLF-MI), should be handled.

High order prediction (HOP) models, e.g., using geometric transformations with more DoF, have been studied during the last two decades in traditional 2D and 3D image coding scenarios. Several geometric models, like translation, rotation, scale, shear and perspective changes have been used to improve the coding efficiency, by exploiting spatial [92], temporal [93]–[98] and inter-view [99]–[102] redundancy. In most proposals, these models have been applied image-wise (instead of block-wise), due to two main reasons: (i) high computational complexity in block-wise model parameter estimation, and (ii) significant additional bit rate required for parameter transmission. Despite these drawbacks, this chapter demonstrates that block-wise HOP models can increase block matching accuracy and, thus, coding efficiency for LF images.

The method proposed in this chapter for encoding LF images relies on a two-stage block-wise HOP model, where each image block is intra predicted from a reference in the causal area of the image, i.e., containing pixels that were already encoded. Since this approach is applied block-wise, it is possible to optimize the HOP model (number of DoF) for each block to be encoded. Taking advantage of the extra DoF available in HOP models, it is possible to outperform state-of-the-art coding techniques based on LOP models.

The remainder of this chapter is organized as follows: Section 5.1 describes the geometric transformations used in the proposed prediction method; Section 5.2 presents the proposed HOP model; Section 5.3 presents the proposed HOP training model; Section 5.4 presents experimental results, and, finally, Section 5.5 concludes the chapter with some final remarks.

5.1. Geometric transformations for high order prediction

In most state-of-the-art encoders, e.g., HEVC, prediction between blocks of pixels is performed using very simple transformations, like translations. However, a lenslet LF image is comprised of MIs that are related by more complex transformations, resulting from the fact that each MI represents the scene being captured from slightly different perspectives. In such cases, it is

advantageous to use, for example, a projective geometric transformation that better exploit the features of the LF image and its MIs.

A geometric transformation (GT) is able to map perspective changes from one view (generically associated to a quadrilateral) into another view, requiring up to eight DoF. Considering two different blocks, A and A', each one with its own coordinate system, (u, v) and (x, y), respectively, it is possible to define a generic relationship:

$$(x, y) = \big(X(u, v), Y(u, v)\big) \tag{5.1}$$

where X and Y are mapping functions for each coordinate. These functions create a point to point correspondence between images. Depending on the number of DoF used by the mapping functions in (5.1), different number of independent point-to-point correspondences are possible. To describe these mapping functions, some GTs may be used, namely, Projective, Bilinear or a simpler Affine GT, as illustrated in Figure 5.1.

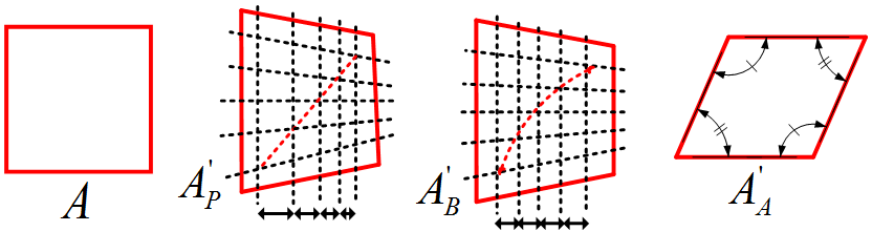


Figure 5.1. Examples of possible GTs applied to block A: Projective (A'_P), Bilinear (A'_B) and Affine (A'_A).

5.1.1. Projective geometric transformation

In order to simplify the mathematics used in this kind of GT, homogeneous coordinates are commonly used [103]. Thus, the Projective GT can be defined by a 3×3 matrix $\boldsymbol{H}$ verifying (5.2):

$$[x, y, 1] = [uh, vh, h]\boldsymbol{H} \tag{5.2}$$

The Projective matrix $\boldsymbol{H}$ can be decomposed into three different submatrices, $\boldsymbol{L_p}$, $\boldsymbol{T_p}$ and $\boldsymbol{P_p}$:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{L_p} & \boldsymbol{P_p} \\ \boldsymbol{T_p} & 1 \end{bmatrix}, \boldsymbol{L_p} = \begin{bmatrix} l_{00} & l_{01} \\ l_{10} & l_{11} \end{bmatrix}, \boldsymbol{T_p} = [t_x \quad t_y], \boldsymbol{P_p^T} = [p_x \quad p_y] \tag{5.3}$$

Each submatrix is responsible for a different elementary type of GT: $\boldsymbol{T_p}$ is responsible for the description of translations, $\boldsymbol{L_p}$ is able to define linear transformations such as rotation, scaling, and shearing, and $\boldsymbol{P_p}$ describes perspective transformations.

To fully exploit the capabilities of the projective matrix $\boldsymbol{H}$, a four-point correspondence is necessary between blocks A and A'. In this case, the full transformation matrix corresponds to the following system of equations:

$$\begin{cases} x = X(u,v) = \dfrac{l_{00}u + l_{10}v + t_x}{p_x u + p_y v + 1} \\ y = Y(u,v) = \dfrac{l_{01}u + l_{11}v + t_y}{p_x u + p_y v + 1} \end{cases} \tag{5.4}$$

The system of equations (5.4) defines the necessary calculations for mapping the coordinates of every pixel of block A into the transformed block A'.

The number of available DoF is directly related with the number of known points of correspondence which exist between both images. For less than four points of correspondence, simpler transformations can be represented by the perspective model. For example, if one point is known, the only component that can be possibly described is a translation, i.e., $\boldsymbol{T_p} = [t_x, t_y]$, $\boldsymbol{P_p} = [0,0]^T$ and $\boldsymbol{L_p} = \boldsymbol{I_2}$, where $\boldsymbol{I_2}$ is the 2×2 identity matrix. This case is defined by (5.5):

$$[x, y, 1] = [uh, vh, h] \begin{bmatrix} \boldsymbol{I_2} & \boldsymbol{0} \\ \boldsymbol{T_p} & 1 \end{bmatrix} \tag{5.5}$$

which can be translated into the system of equations (5.6):

$$\begin{cases} x = X(u) = u + t_x \\ y = Y(v) = v + t_y \end{cases} \tag{5.6}$$

This is the case in most state-of-the-art MI search-based schemes to encode LF images [50], where one or several vectors signal one or several 2D displacements, i.e., using only two DoF.

5.1.2. Bilinear geometric transformation

The Bilinear GT is an alternative to the Projective GT, defined by a 4×2 matrix $\boldsymbol{B}$ verifying (5.7):

$$[x, y] = [uv, u, v, 1]\boldsymbol{B}, \tag{5.7}$$

where:

$$\boldsymbol{B} = \begin{bmatrix} \boldsymbol{P_b} \\ \boldsymbol{L_b} \\ \boldsymbol{T_b} \end{bmatrix} = \begin{bmatrix} p_x & p_y \\ l_{00} & l_{01} \\ l_{10} & l_{11} \\ t_x & t_y \end{bmatrix} \tag{5.8}$$

The Bilinear GT matrix $\boldsymbol{B}$ can represent similar GTs as the Projective GT, with the same number of DoF. However, it performs a non-planar transformation, which makes it more flexible. Thus, only horizontal and vertical lines, as well as equispaced points along these directions, are preserved [104]. Diagonal lines, on the other hand, are not mapped as lines but as quadratic curves. This feature is illustrated in Figure 5.1, where, in the case of the Bilinear GT, points along vertical parallel lines are kept equispaced, while the points along diagonal lines are mapped onto a quadratic curve (see block A'_B). When the Projective GT (block A'_A) is used, points along the parallel vertical lines do not stay equispaced but points along diagonal lines are also mapped along a line. Another property of this GT, when compared to the Projective GT, is the need for simpler calculations per pixel, given by (5.9):

$$\begin{cases} x = X(u, v) = uvp_x + ul_{00} + vl_{10} + t_x \\ y = Y(u, v) = uvp_y + ul_{01} + vl_{11} + t_y \end{cases}, \tag{5.9}$$

5.1.3. Affine geometric transformation

When using either the Projective or the Bilinear GT, eight DoF are available. However, a simpler case exists, which is known as the Affine GT, that can describe GTs up to six DoF. The Affine GT can be described as a particular case of Projective or Bilinear GTs, by using matrices $\boldsymbol{H}$ and $\boldsymbol{B}$ with $\boldsymbol{P_p^T} = [0 \quad 0]$ and $\boldsymbol{P_b} = [0 \quad 0]$, respectively. This GT only requires three points of correspondence between images, defined by (5.10):

$$\begin{cases} x = X(u, v) = l_{00}u + l_{10}v + t_x \\ y = Y(u, v) = l_{01}u + l_{11}v + t_y \end{cases}, \tag{5.10}$$

5.2. Proposed high order prediction mode

This section proposes a LF image coding method, based on a HOP mode, which is implemented as a block-wise prediction mode in HEVC. This HOP mode is added to the set of HEVC intra prediction modes, i.e., DC, Planar and the 33 intra Directional modes.

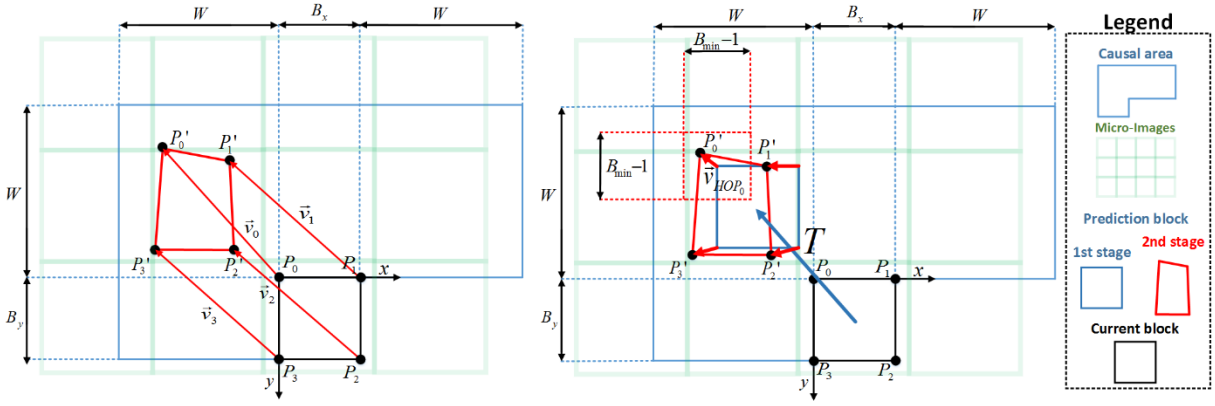


Figure 5.2. Block prediction using a HOP model: generic single-stage HOP model mapping (left side), and proposed two-stage HOP model mapping (right side).

The proposed HOP mode predicts each block by applying a GT between two quadrilaterals, the current block and a block in the reference region, the causal area of pixels already encoded. The algorithm for the proposed prediction mode can be described through the following steps:

1. **Selection of the next set of correspondence points to be evaluated**: Selection of a quadrilateral in the causal area of pixels (from a set of pre-defined cases), with corners $\{P_n'\}$, that is mapped into the block which is being predicted, with corners $\{P_n\}$ (see left side of Figure 5.2);

2. **Calculation of the GT parameters**: Calculation of the transformation parameters that map the quadrilateral defined by $\{P_n'\}$ into the one defined by $\{P_n\}$;

3. **Inverse GT mapping**: Mapping of the causal quadrilateral defined by $\{P_n'\}$ to the one defined by $\{P_n\}$, using an inverse mapping procedure with the parameters calculated in the previous step, in order to compute the block prediction error and the estimated number of bits to transmit the GT parameters;

4. **Estimation of the GT RD cost**: Estimation of the rate-distortion (RD) cost, J, associated to the GT that is being evaluated, considering the computed block prediction;

5. **Repeat the above steps to find the GT with minimum RD cost***:* Evaluate iteratively all the pre-defined combinations of correspondence points and choose the one that has the minimum RD cost J.

6. **Encode the HOP mode information***:* The corner displacement between the quadrilateral in the causal area $\{P_n'\}$ and the block which is being predicted $\{P_n\}$ is signaled to the decoder.

The following sections explain each step of the proposed HOP mode in more detail.

5.2.1. Selection of the correspondence points

The major challenges faced by the proposed algorithm are the computational complexity required to estimate the optimal set of GT parameters and the necessary bitrate for transmitting these data. To tackle both problems, a rate-distortion-complexity tradeoff is defined. From Figure 5.2 (left side) it can be inferred that, if all possible four-point correspondences between the prediction block and the current block to be encoded were evaluated, the number of tested transformations per block would be larger than $(2W^2)^4$, i.e., for a search window ($W = 128$) more than 1.15×10^{18} correspondence possibilities per block exist. To reduce the number of tests to a practicable number, a two-stage minimization problem is proposed, aiming to determine a good approximation to the optimal HOP model, as illustrated in Figure 5.2 (right side):

LOP Model Estimation

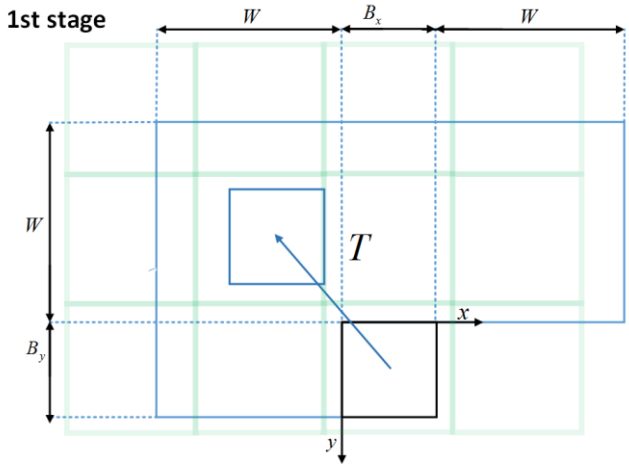


Figure 5.3. LOP model estimation.

In the first stage, a pure translational LOP model (two DoF) is used as shown in Figure 5.3. The result of this stage is, the bidimensional vector, ***T***, with the lowest RD cost, pointing into the search window of the causal area (see the blue vector on the right side of Figure 5.2). The search to determine ***T*** is performed using a full search algorithm, as described in [50]. The LOP estimation stage of the proposed HOP mode is based on the SS prediction method. The prediction cost is minimized by testing all the possible positions inside the search window for a single vector that relates the current block to the prediction block. The ***T*** vectors, generated by the first stage, can be either encoded explicitly, similarly to motion vectors in HEVC or using the SS-Skip mode, which creates a list of candidates that includes the ***T*** vectors used to encode neighboring blocks. If one candidate from this list is selected to encode the current block, it is only necessary to encode its index, as in the HEVC merge mode. Additionally, some predetermined vectors are added to the candidate list, referred to as MI-based candidates [50]. These candidates correspond to vectors that are very likely to be selected by the SS prediction mode, such as, vectors pointing to the same spatial position of the current MI within the left, above and above-left MIs.

HOP Model Estimation

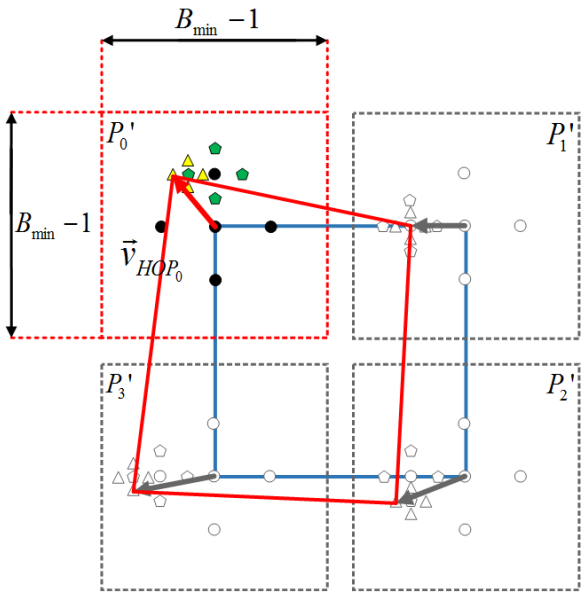


Figure 5.4. Fast search method adopted for each corner of the prediction block (blue rectangle) used to estimate the HOP model (red quadrilateral).

In the second stage, a HOP model (up to eight DoF) is used, employing as a starting point the result of the first stage (see, respectively, the red and blue quadrilaterals on the right side of

Figure 5.2). For this, a set of four vectors, $\{v_{HOP_n}\}$, is computed, each of them defining the position of one corner of the reference quadrilateral, thus defining the 2D GT.

To further reduce the computational complexity of the second stage of this minimization problem, a 2D logarithmic fast search method has been adopted, which is applied to each corner of the prediction block (blue rectangle). In this case, the maximum number of search steps has been set to $log_2(min(B_x, B_y)) - 1$, depending on the size of the prediction block, i.e., B_x (width) and B_y (height). In each step, the searching points are defined according to a five-point small diamond-shaped basis pattern with an initial search step size equal to $min(B_x, B_y)/4$ [105], [106]. This 2D logarithmic fast search method using the five-point small diamond-shaped basis pattern is graphically represented in Figure 5.4 across three search steps, represented, respectively, by black circles, green pentagons and yellow triangles. After each search step, the point that minimizes the RD cost function is set as the center of the next step and the search step size is halved until a unitary step value is reached. In the example of Figure 5.4, in the first corner (P_0), the five points associated with the first step, represented by the black circles, are tested. The point that minimizes the RD cost function for the first search step is the black circle on the top. For the second and third search steps, the points on left, respectively, green pentagon and yellow triangle, are the points that yield the lowest RD cost. The final point is selected to define the red arrow that describes the corner displacement of the first corner of the block.

Considering that the search procedure must be applied to all the corners of the prediction block over several search steps, there are two ways of implementing this second stage search: by jointly optimizing each step of the search procedure for the four corners or by independently optimizing each step of the search procedure. By considering five points for each of the S search steps of the 2D logarithm search, the required number of search points for each option is given by $(5 \times S)^n$ or $5^n \times S$, respectively where n is the number of corners. In order to reduce the computational complexity, the second option was used, where each step is optimized individually.

The stop condition for this search method is met when the corner step size reaches the unit. Therefore, the example shown in Figure 5.4 represents the unitary steps as the yellow triangles. Since the underlying codec uses variable block sizes, S will depend on the block size. The search

window for each corner is limited to $min(B_x, B_y) - 1$, as illustrated in Figure 5.4 (see the dashed red block).

The quadrilateral used by the HOP model estimation may be scaled to increase pixel precision. Figure 5.2 and Figure 5.4 illustrate the second stage applied to a blue rectangle with the same size of the block being predicted (in black) to not overload the figures. However, in terms of our implementation, a rectangle, twice the size of the original block, is used to determine the HOP model. This modification means that an integer pixel displacement in one of the corners of the large quadrilateral corresponds to a sub-pixel displacement in the area of the original rectangle. For a block twice the size of the original block, one extra search step is performed by the 2D logarithmic search algorithm that is used at the HOP stage. This occurs because the stopping condition for the search algorithm is the unitary step size. The pixel precision can be further extended by using a rectangle with sides four or eight times the size of the original blue rectangle, which increase, the number of search steps by one or two, respectively. After extensive testing, the best solution in a RD sense was adopted, that is increasing the blue rectangle to twice the original size, despite requiring one extra step.

As the second stage of the HOP search can be biased by the first stage result, the global result of the search method also tests the $\boldsymbol{T}$ vectors used in the previously encoded neighboring blocks (vector predictors), instead of considering only the best $\boldsymbol{T}$ vector from the first stage. Additionally, other $\boldsymbol{T}$ vectors can be tested in conjunction with the HOP model estimation, e.g., top ten candidates from the first stage. However, it was experimentally verified in this Thesis that the vectors that are more RD cost efficient are the $\boldsymbol{T}$ vector predictors.

The proposed approach can be implemented using either the Projective GT defined in (5.3):

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{T} & 0 \end{bmatrix} + \begin{bmatrix} \boldsymbol{L'_p} & \boldsymbol{P'_p} \\ \boldsymbol{T'_p} & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{L'_p} & \boldsymbol{P'_p} \\ \boldsymbol{T} + \boldsymbol{T'_p} & 1 \end{bmatrix}, \tag{5.11}$$

or the Bilinear GT defined in (5.8):

$$\boldsymbol{B} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{T} \end{bmatrix} + \begin{bmatrix} \boldsymbol{P'_b} \\ \boldsymbol{L'_b} \\ \boldsymbol{T'_b} \end{bmatrix} = \begin{bmatrix} \boldsymbol{P'_b} \\ \boldsymbol{L'_b} \\ \boldsymbol{T} + \boldsymbol{T'_b} \end{bmatrix}, \tag{5.12}$$

Where $\boldsymbol{T}$ is the vector estimated during the LOP stage and $\boldsymbol{T'}$, $\boldsymbol{L'}$ and $\boldsymbol{P'}$ are the GT parameters that describe the HOP stage.

5.2.2. Calculation of the GT parameters

After obtaining vector $\boldsymbol{T}$ (see the right side of Figure 5.2), it is possible to determine submatrices $\boldsymbol{T}'$, $\boldsymbol{P}'$ and $\boldsymbol{L}'$ in equations (5.11) and (5.12), by using their width and height, B_x and B_y, respectively, and the small vectors associated with the corner position change of the blue rectangle:

$$\begin{cases} \vec{v}_{HOP_0} = (u_0, v_0) \\ \vec{v}_{HOP_1} = (u_1 - (B_x - 1), v_1) \\ \vec{v}_{HOP_2} = (u_2 - (B_x - 1), v_2 - (B_y - 1))' \\ \vec{v}_{HOP_3} = (u_3, v_3 - (B_y - 1)) \end{cases} \tag{5.13}$$

Note that in the proposed two-stage approach, vectors $\vec{v}_n$, represented in the left of Figure 5.2, correspond to the sum of vector $\boldsymbol{T}$ from the first stage, with the four smaller vectors from the second stage, i.e.,$\vec{v}_n = T + \vec{v}_{HOP_n}$, represented in the right of Figure 5.2. If the Projective GT is used some auxiliary variables are defined:

$$\begin{cases} \Delta u_1 = u_1 - u_2 \\ \Delta u_2 = u_3 - u_2 \\ \Delta u_3 = u_0 - u_1 + u_2 - u_3 \end{cases} \quad \begin{cases} \Delta v_1 = v_1 - v_2 \\ \Delta v_2 = v_3 - v_2 \\ \Delta v_3 = v_0 - v_1 + v_2 - v_3 \end{cases}, \tag{5.14}$$

The Affine GT can be defined by any three of the four vectors ($\vec{v}_{HOP}$). In this Thesis, the first three vectors, $\vec{v}_{HOP_0}$, $\vec{v}_{HOP_1}$and $\vec{v}_{HOP_2}$ are generated using the second stage of the proposed approach, where the remaining vector, $\vec{v}_{HOP_3}$, is calculated assuming $\Delta u_3 = \Delta v_3 = 0$, thus resulting in $\vec{v}_{HOP_3} = (u_0 - u_1 + u_2,\ v_0 - v_1 + v_2 - (B_y - 1))$.Using (5.14), the individual parameters in the submatrices can then be calculated by (5.15) for the Projective GT:

$$\boldsymbol{P}'_{\boldsymbol{p}} = \begin{bmatrix} \frac{1}{B_x - 1} \begin{vmatrix} \Delta u_3 & \Delta u_2 \\ \Delta v_3 & \Delta v_2 \end{vmatrix} & \frac{1}{B_x - 1} \begin{vmatrix} \Delta u_1 & \Delta u_3 \\ \Delta v_1 & \Delta v_3 \end{vmatrix} \\ \frac{1}{B_y - 1} \begin{vmatrix} \Delta u_1 & \Delta u_2 \\ \Delta v_1 & \Delta v_2 \end{vmatrix} & \frac{1}{B_y - 1} \begin{vmatrix} \Delta u_1 & \Delta u_2 \\ \Delta v_1 & \Delta v_2 \end{vmatrix} \end{bmatrix} \quad \boldsymbol{L}'_{\boldsymbol{p}} = \begin{bmatrix} \frac{u_1 - u_0}{B_x - 1} + p_x u_1 & \frac{u_3 - u_0}{B_y - 1} + p_y u_3 \\ \frac{v_1 - v_0}{B_x - 1} + p_x v_1 & \frac{v_3 - v_0}{B_y - 1} + p_y v_3 \end{bmatrix} \tag{5.15}$$

Similarly, for the Bilinear GT, the corresponding submatrices are calculated by (5.16):

$$\boldsymbol{P_b'^T} = \begin{bmatrix} \frac{u_0-u_1+u_2-u_3}{(B_x-1)(B_y-1)} \\ \frac{v_0-v_1+v_2-v_3}{(B_x-1)(B_y-1)} \end{bmatrix} \quad \boldsymbol{L_b'} = \begin{bmatrix} \frac{u_3-u_0}{B_y-1} & \frac{v_3-v_0}{B_y-1} \\ \frac{u_1-u_0}{B_x-1} & \frac{v_1-v_0}{B_x-1} \end{bmatrix}. \tag{5.16}$$

For both cases:

$$\boldsymbol{T_p'} = \boldsymbol{T_b'} = [u_0 \quad v_0]. \tag{5.17}$$

5.2.3. Inverse GT mapping

As previously mentioned, a GT between two blocks corresponds to a mapping of every pixel within one block into the other block, e.g., the mapping functions (5.4) and (5.9) correspond to the Projective and Bilinear GT, respectively. When the mapping is performed from the rectangular block to be encoded to an arbitrary reference quadrilateral it is called a direct mapping, otherwise it is called an inverse mapping. An example of both mapping procedures for a simple scaling GT can be found in Figure 5.5. As can be observed in Figure 5.5, when direct mapping is used the final quadrilateral shape (red block) does not match the desired reference block pixel grid, requiring to perform pixel interpolation prior to calculate the distortion between the transformed block and the reference block. For the sake of simplicity, an inverse mapping has been adopted, as it generates a rectangular prediction block with the same dimensions of the block to be encoded.

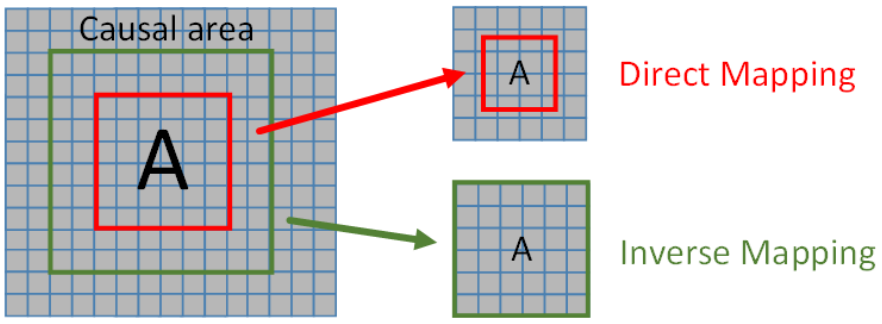


Figure 5.5. Example of Direct Mapping and Inverse Mapping when a scale GT is applied.

Thus, regardless of the size of the quadrilateral used for estimation, (5.4) and (5.9) take as input the coordinates of the block to be encoded, i.e., $u \in [0, B_x - 1]$ and $v \in [0, B_y - 1]$, and generate as output the coordinates (x, y) in the causal area, where the reference pixel value is going to be extracted from. Since x and y are typically fractional values, a bilinear interpolation filter is used to compute the actual pixel value.

5.2.4. Estimation of the GT RD cost

The optimal HOP model for each block is determined through RD optimization, minimizing the associated Lagrangian cost, $J = D + \lambda R$, over the entire set of pre-defined GT. D refers to the distortion between the prediction block and the current block, R is the estimated number of bits used to encode the block using the GT under evaluation, and λ is the Lagrange multiplier. The parameter λ is the same for all prediction modes, including the intra modes, so no biases in terms of prediction mode selection are introduced. In this Thesis, D, is computed as the sum of absolute differences (SAD) in the pixel domain, in the first stage, and SAD in the Hadamard domain, in the second stage, as suggested in [107].

By using a two-stage method it is possible to evaluate if it is more advantageous to use LOP or HOP for each block, by comparing the associated costs, given by:

$$J_{LOP} = D_{LOP} + \lambda R_{LOP} \text{ and } J_{HOP} = D_{HOP} + \lambda R_{HOP}, \tag{5.18}$$

where R_{LOP} and R_{HOP} are the estimated number of bits for the corresponding coding mode.

The usage of LOP or HOP is conveyed to the decoder through a binary flag, F_{HOP}. When LOP is considered more efficient in a RD sense, $F_{HOP} = 0$, and only $\boldsymbol{T}$ is transmitted in the bitstream. On the contrary, if HOP is used, all the elements that describe $\boldsymbol{T}$ are transmitted, followed by $F_{HOP} = 1$ and the four additional $\vec{v}_{HOP_n}$ vectors.

The number of bits required to signal the HOP mode, R_{HOP}, is the sum of R_{LOP} and the estimated bits for encoding the four vectors, $\vec{v}_{HOP_n}$, that define the used HOP model. The rate of these small amplitude vectors is estimated using the same procedure as vector $\boldsymbol{T}$.

5.2.5. Encode the HOP mode information

After finding the optimal HOP model, the cost of the HOP mode, J_{HOP}, is compared against the cost of the other intra prediction modes, i.e., DC, Planar and the 33 Directional modes, and the mode with the lowest RD cost is encoded. For this, CABAC entropy coding method is used by HEVC is used to encode the HOP mode information. The CABAC entropy coder is based on three steps: (i) binarization of syntax elements, (ii) context modeling, and (iii) binary arithmetic coding. In this implementation, these three steps have been maintained using, however, new contexts. Vectors $\boldsymbol{T}$ and $\vec{v}_{HOP_n}$, and flag, F_{HOP}, are transmitted to the decoder using the HEVC

approach for motion vectors and merge flags [108]. To encode $\boldsymbol{T}$, the same syntax elements of HEVC for motion data are used, i.e., motion vector differences, motion vector prediction index, RPL and RPL index.

The way the HOP model information is conveyed to the decoder can highly influence the coding efficiency. One possible approach is to send the GT parameters, i.e., in the $\boldsymbol{H}$ or $\boldsymbol{B}$ matrix, which need to be represented with high precision. Alternatively, as proposed in this Thesis, the encoder just sends the four vectors, $\vec{v}_{HOP_n}$, which can be represented with just a few bits. The major advantage of encoding the GT parameters matrix is that they do not need to be recalculated at the decoder side through equations (5.13) – (5.17). However, they need to be encoded with a very high precision because these values are not very robust to quantization [96]. Consequently, encoding the vectors, $\vec{v}_{HOP_n}$, leads to higher compression efficiency

5.3. Proposed high order prediction training

This section proposes an alternative LF image coding method to the method proposed in section 5.2 also based on the same HOP model. This alternative HOP method is able to estimate an efficient GT using a HOP training stage applied to a causal area of the LF image. The same two-stage block-wise HOP model is used, however, instead of explicitly transmitting the HOP vectors, i.e., $\vec{v}_{HOP_n}$., the vectors are inferred by the decoder. In this case, a HOP training is applied using several predetermined training directions in the causal area of the LF image. Each training direction corresponds to the location of adjacent MIs that are already encoded, e.g., the upper, left and upper-left MIs. Since adjacent MIs are typically very similar, it is expected that the GT that is generated from the training step, will also produce an efficient prediction block. The most efficient training direction is transmitted to the decoder as an index. Since the training is performed in the causal area of the LF image, the GT for the second stage of the proposed HOP approach can be also calculated at the decoder side. Therefore, no overhead is necessary to describe the second stage of the HOP approach, but the most efficient training direction index.

The proposed HOP training algorithm can be described by the following steps:

1. **LOP model estimation:** The LOP model is applied in order to estimate $\boldsymbol{T}$ vector and generate prediction block B_{LOP} described in Section 5.2.1;

2. $\boldsymbol{J_{LOP}}$ **cost estimation:** The J_{LOP} cost is estimated which accounts for transmitting the LOP model information, described in Section 5.2.4;
3. **HOP model estimation for each $\boldsymbol{n}$ predetermined training direction**:
    i. **HOP model estimation**: The HOP model explained in Section 5.2 is used with the n prediction blocks B_t and the reference region to generate a GT_n candidate
    ii. **Apply the GT candidate to $\boldsymbol{B_{LOP}}$**: The output from the HOP model, i.e., the GT candidate, is applied to B_{LOP} in order to generate the prediction block, B_{HOP_n}
    iii. $\boldsymbol{J_{HOP}}$ **cost estimation**: The J_{HOP_n} cost is estimated, which accounts for transmitting the HOP information necessary to generate B_{HOP_n}, i.e., vector $\boldsymbol{T}$ and HOP training index $(n+1)$
4. **LOP and HOP parameters encoding**: Encode the information for both LOP and HOP that corresponds to the lowest cost among J_{LOP} and J_{HOP_n} candidates, respectively.

In the first stage, a LOP search is used between the current block and the reference region. A full search algorithm is used, as proposed in Section 5.2.1, and the output is a translational vector $\boldsymbol{T}$ (two DoF). This option is available in case the HOP training is not able to find a "good" GT candidate. The efficiency of the HOP training tends to be higher when the redundancy between the current block and at least one of the B_t blocks, available in each training direction, is high.

The proposed algorithm can be applied to any number of training directions as it is shown in Figure 5.6. The goal is to find a training direction that minimizes the RD cost of the generated prediction block, B_{HOP_n}. Since the HOP training index is transmitted to the decoder, only the training that provides the best result is repeated on the decoder side. However, the number of bits necessary to transmit the HOP training index increases with the number of training directions. The training directions are selected based on the proximity to the current block. For example, for three training directions, i.e., $n = 3$, using a square-based MLA, the blocks B_t are located in: $B_{t_0} = (-m, 0)$; $B_{t_1} = (0, -m)$, which are both shown in Figure 5.6, and $B_{t_2} = (-m, -m)$. Where m is the size of the MI in pixels. These locations correspond to estimated locations of the block B_{t_n} in the left, upper and upper-left MIs.

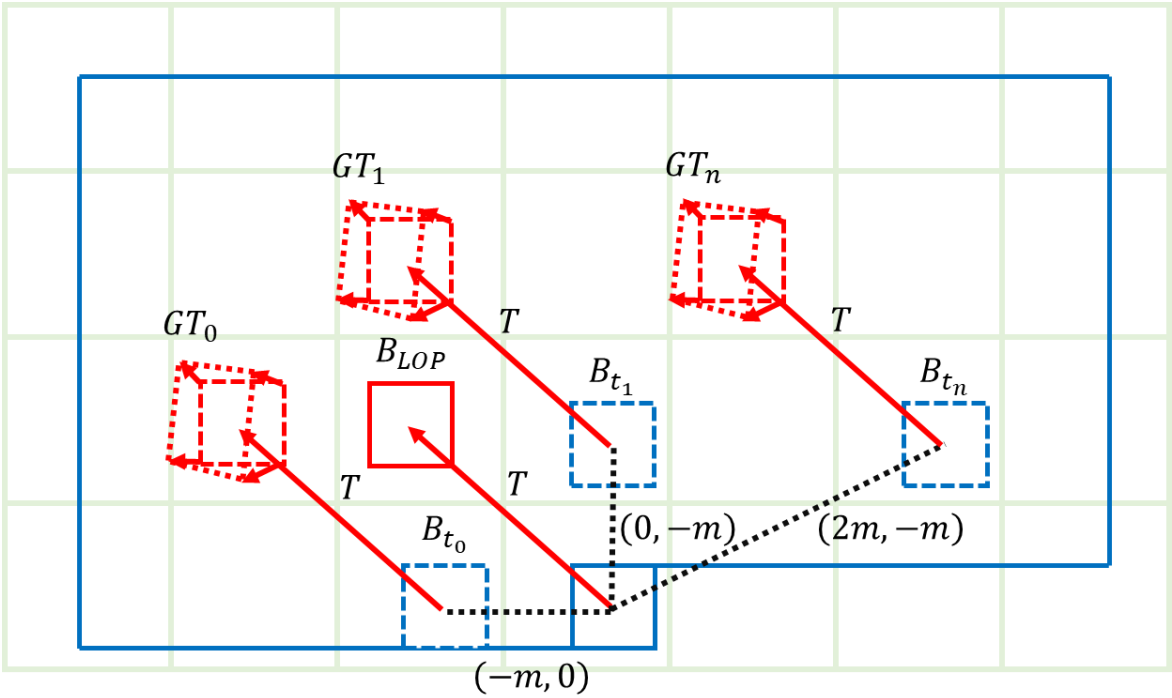


Figure 5.6. Proposed HOP training algorithm being applied in three different directions, left, upper and upper-right direction.

For each of the defined training directions, the HOP model is applied (see Section 5.2), using as an input block B_{t_n} and the reference region. The output of the HOP model estimation is a GT candidate per training direction. Each GT candidate is therefore applied to the prediction block, B_{LOP}, generating n different B_{HOP_n} prediction blocks. To determine which training direction is the one that generates the most efficient prediction block, an RD cost value is calculated for each of the n training directions.

The RD cost is mostly calculated the same way as in Section 5.2.4, with the only exception being the rate associated with the HOP prediction, R_{HOP}. In this case, the estimated R_{HOP} is given by $\log_2(n+1)$, where n is the number of training directions, because instead of transmitting the individual HOP vectors, the training direction index is transmitted.

Once all the costs J_{LOP} and J_{HOP_n} associated to each possible candidate are generated, the lowest one corresponds to the most RD efficient. The most efficient option is then compared to the cost of the other intra prediction modes, i.e., DC, Planar, and the 33 Directional modes, and the mode with the lowest RD cost is selected and encoded. Vector $\boldsymbol{T}$ and the HOP training index are

transmitted to the decoder using the HEVC approach for motion vectors and motion vector prediction index, and encoded using the CABAC entropy coding method.

5.4. Experimental results

In this section the performance of the proposed LF coding solution, incorporating the HOP mode and HOP training, is evaluated in comparison with state-of-the-art coding solutions based on LOP approaches. First, this section describes the test conditions, including the used LF test images, the benchmark solutions and the relevant test parameters. Afterwards, experimental results comparing the RD performance of different types of prediction models and evaluation processing chains are presented and discussed. These results are complemented with some statistical information about prediction mode usage.

5.4.1. Test conditions

In order to evaluate the RD performance of the proposed LF coding solution, two types of LF images were selected for the experimental test setup. The first type of images were acquired using LF cameras with a focused (FOC) optical setup (see Appendix A.1), which include two images from the *Plane and Toy* sequence; one image from *Demichelis Spark* and *Demichelis Cut*; and *Laura* and *Seagull* LF images. The second type of images were acquired using a Lytro Illum camera that is commercially available and uses an unfocused (UNF) optical setup extracted from the EPFL LF dataset (see Appendix A.2). All images use the YUV 4:2:0 8 bit color format. This selection includes LF images with different resolutions, MI resolutions and types of microlens arrays, with different MI shape and different LF camera models.

Two evaluation processing chains are used for testing, which include the BEPC, presented in Section 2.6.1 when encoding the LL images directly (both UNF and FOC LF images), and the JPEG Pleno CTCs, presented in Section 2.6.2, when encoding the UNF LF images using the 4DLF-MI data representation. The FOC LF images are not tested using the JPEG Pleno CTCs because no metadata is included with the LF images, which hampers the conversion to the 4D LF data representation.

The solution using the proposed HOP mode will be referred to as HEVC-HOP, while the solution using the proposed HOP training will be referred as HEVC-HOP-nT, where n is the

number of predetermined training directions. Finally, HEVC using only the standard intra modes is simply referred to as HEVC. Additionally, the work in [50] is used as benchmark for RD performance of a LOP-based solution and it is referred to as HEVC-SS.

All the test images were encoded using QP values of 22, 27, 32, 37, 42 and 47. The causal window size W is 128 for both HEVC-HOP, HEVC-HOP-nT and HEVC-SS, for every encoded image. The number of available SS or $\boldsymbol{T}$ vector predictors, used for coding, is two. These $\boldsymbol{T}$ vector predictors are used as additional vector $\boldsymbol{T}$ candidates for the LOP model estimation stage. The number of candidates available for SS-Skip is five in both HEVC-SS and HEVC-HOP and HEVC-HOP-nT. All the referred LF images are encoded and decoded using the HEVC, HEVC-HOP, HEVC-HOP-nT and HEVC-SS codecs, and the RD performance is evaluated using a Bjøntegaard delta (BD) Metrics [109].

5.4.2. HOP mode performance assessment

When evaluating the HOP mode, several variants of HEVC-HOP are tested. These variants of HEVC-HOP are referred to as HEVC-HOP-A, HEVC-HOP-P and HEVC-HOP-B, respectively for Affine (6 DoF), Projective (8 DoF) and Bilinear (8 DoF) GTs. Table 5.1 shows the RD performance comparison between HEVC and HEVC-SS, and between HEVC-SS and the various HEVC-HOP variants using the BEPC (see Section 2.6.1). The values in bold correspond to the best achieved result for each specific test image.

Comparison between LOP and HOP

Table 5.1 shows that HEVC-SS can outperform HEVC, for all tests, with bitrate savings up to 45.35%. Nevertheless, all versions of the proposed HEVC-HOP method are even more efficient than HEVC-SS to encode LF images. This increased performance, with bitrate savings up to 12.62% for certain images relatively to HEVC-SS (49.82% relatively to HEVC), comes from the use of a higher order prediction model. Since HEVC-SS is limited to two DoF, it is not able to accurately describe block transformations more complex than a simple translation. When comparing the results by means of comparing the effectiveness of adding prediction tools with more than two DoF, it is possible to notice that for the encoded LF images, the best case is when eight DoF are used. If eight DoF are available, i.e., when HEVC-HOP-P is being used, four points of correspondence are transmitted, which allows the description of not only translations,

but also rotations, scaling, shearing and perspective changes. In this case, although extra information needs to be encoded, relative to the HEVC-SS case, the bitrate savings increases to 5.86% (28.81% relative to HEVC), on average, for all tested LF images.

Table 5.1. BD-PSNR-Y and BD-Rate results using the BEPC comparing HEVC, HEVC-SS (2 DoF) and HEVC-HOP, using 6 DoF and 8 DoF and two different geometric transformations

Image	HEVC-SS (2 DoF) vs HEVC		HEVC-HOP-A (6 DoF) vs HEVC-SS (2 DoF)		HEVC-HOP-P (8 DoF) vs HEVC-SS (2 DoF)		HEVC-HOP-B (8 DoF) vs HEVC-SS (2 DoF)	
	BD-PSNR-Y	BD-RATE	BD-PSNR-Y	BD-RATE	BD-PSNR-Y	BD-RATE	BD-PSNR-Y	BD-RATE
PT0	0.90 dB	-14.64 %	0.23 dB	-3.93 %	**0.27 dB**	**-4.66 %**	0.23 dB	-4.02 %
PT150	1.44 dB	-19.02 %	0.64 dB	-9.44 %	**0.75 dB**	**-11.05 %**	0.71 dB	-10.50 %
DS	1.09 dB	-31.43 %	0.23 dB	-7.41 %	**0.26 dB**	**-8.39 %**	0.26 dB	-8.31 %
DC	1.05 dB	-29.25 %	0.26 dB	-8.06 %	**0.30 dB**	**-9.14 %**	0.29 dB	-8.83 %
Laura	2.26 dB	-30.35 %	0.15 dB	-2.76 %	0.27 dB	-4.78 %	**0.32 dB**	**-5.62 %**
Seagull	2.81 dB	-42.78 %	0.22 dB	-4.89 %	0.31 dB	-6.82 %	**0.43 dB**	**-9.21 %**
I01	0.81 dB	-18.50 %	0.11 dB	-2.91 %	**0.13 dB**	**-3.34 %**	0.11 dB	-2.97 %
I02	0.61 dB	-14.67 %	0.10 dB	-2.72 %	**0.11 dB**	**-2.94 %**	0.10 dB	-2.75 %
I03	0.17 dB	-4.10 %	0.03 dB	-0.75 %	**0.03 dB**	**-0.77 %**	0.03 dB	-0.67 %
I04	0.25 dB	-6.31 %	**0.02 dB**	**-0.61 %**	0.02 dB	-0.57 %	0.02 dB	-0.57 %
I05	0.89 dB	-28.73 %	0.10 dB	-3.90 %	**0.14 dB**	**-5.30 %**	0.12 dB	-4.89 %
I06	1.43 dB	-45.35 %	0.05 dB	-1.26 %	**0.06 dB**	**-4.46 %**	0.05 dB	-2.91 %
I07	0.48 dB	-13.73 %	0.26 dB	-8.05 %	0.25 dB	-7.78 %	**0.29 dB**	**-8.93 %**
I08	0.66 dB	-22.95 %	0.06 dB	-3.52 %	**0.06 dB**	**-5.19 %**	0.05 dB	-2.62 %
I09	1.49 dB	-30.72 %	0.17 dB	-4.39 %	**0.20 dB**	**-5.31 %**	0.20 dB	-5.28 %
I10	0.22 dB	-8.10 %	0.05 dB	-2.01 %	0.06 dB	-2.21 %	**0.06 dB**	**-2.36 %**
I11	1.49 dB	-42.84 %	0.27 dB	-11.19 %	0.31 dB	-12.30 %	**0.32 dB**	**-12.62 %**
I12	1.42 dB	-41.35 %	0.24 dB	-9.28 %	0.27 dB	-10.42 %	**0.27 dB**	**-10.47 %**
AVG. FOC	1.59 dB	-27.91 %	0.29 dB	-6.08 %	0.36 dB	-7.47 %	**0.37 dB**	**-7.75 %**
AVG. UNF	0.83 dB	-23.11 %	0.12 dB	-4.22 %	**0.14** dB	**-5.06 %**	0.14 dB	-4.75 %
AVG. ALL	1.08 dB	-24.71 %	0.18 dB	-4.84 %	**0.21** dB	**-5.86 %**	0.21 dB	-5.75 %

Comparison between the proposed GTs

The proposed prediction mode HEVC-HOP-B, using a Bilinear GT, can achieve similar results to HEVC-HOP-P for most images, both in terms of average PSNR (BD-PSNR) and bitrate savings. However, comparing the average performance of each method regarding the type of camera models (AVG. FOC and AVG. UNF) it is possible to observe that HEVC-HOP-P is slightly more efficient for the UNF model images and HEVC-HOP-B is slightly more efficient for the FOC model images. In the case of HEVC-HOP-A only six DoF are available because only three points of correspondence are transmitted. When compared HEVC-HOP-A to HEVC-HOP-P, the bitrate savings gains relatively to HEVC-SS are reduced to 4.84% (28.12%

relatively to HEVC) on average considering all tested LF images, which may be due to the fact that HEVC-HOP-A is not able to compensate for perspective changes. However, in terms of computational complexity HEVC-HOP-A is approximately 4.5 times faster than HEVC-HOP-P. Note that none of the implementations is optimized in terms of computational complexity; therefore, the reported values for comparison may vary. It is worthwhile mentioning that for some cases (e.g., I04 test image) HEVC-HOP-A can outperform both eight DoF GTs. In HEVC-HOP-P four correspondence points are always encoded, even if only three are necessary. Since in some cases, more information might be transmitted to describe the same GT, HEVC-HOP-P is, for this particular test image, less efficient than HEVC-HOP-A.

Regarding the computational complexity, a study was performed using the image VESPA, from the EPFL LF dataset. This image was encoded and decoded using the codecs, HEVC, HEVC-SS, HEVC-HOP-A, HEVC-HOP-P and HEVC-HOP-B, with QP=32. These tests were performed using a PC equipped with an Intel Xeon CPU E3-1240 V2@3.4GHz and 24GB of RAM, running Ubuntu 16.04. The obtained running time to encode and decode each image is depicted in Table 5.2. The computational complexity of the proposed schemes must be compared to HEVC-SS, as it is used as reference. The HEVC-SS complexity is equivalent to encoding a P-Slice in HEVC [50]. As can be seen from Table 5.2, the proposed algorithm increases the computational burden at the encoder side, where HEVC-HOP-A, HEVC-HOP-P and HEVC-HOP-B are 1.51, 6.84 and 5.23 times more complex than HEVC-SS, respectively. However, at the decoder the running time is reduced in relation to HEVC-SS. As the proposed method uses a more efficient prediction, the LF image encoder creates a lower number of partitions than the HEVC-SS.

Table 5.2. Codec computational complexity comparison

Encoder	HEVC	HEVC-SS	HEVC-HOP-A	HEVC-HOP-P	HEVC-HOP-B
Run time (h)	0.06	3.88	5.88	26.54	20.30
vs HEVC-SS	0.02	1	1.51	6.84	5.23
Decoder	HEVC	HEVC-SS	HEVC-HOP-A	HEVC-HOP-P	HEVC-HOP-B
Run time (s)	1.70	33.56	30.54	29.98	28.56
vs HEVC-SS	0.05	1	0.91	0.89	0.85

Table 5.3. Average prediction mode usage in percentage of pixels for the HEVC-HOP-P case

Image	DC, Planar and Directional	Proposed prediction method		SS-Skip [50]
		LOP stage	HOP stage	
PT0	**57.76 %**	3.32 %	22.81 %	16.12 %
PT150	26.53 %	4.39 %	**51.27 %**	17.62 %
DS	27.35 %	3.84 %	**40.31 %**	28.51 %
DC	27.25 %	1.93 %	**44.04 %**	26.78 %
Laura	21.51 %	10.25 %	**45.22 %**	23.03 %
Seagull	13.87 %	9.27 %	**41.81 %**	35.05 %
I01	**44.12 %**	5.69 %	33.44 %	16.75 %
I02	**49.28 %**	5.99 %	31.15 %	13.59 %
I03	**81.54 %**	2.63 %	10.55 %	5.28 %
I04	**82.66 %**	2.75 %	9.60 %	4.99 %
I05	**38.42 %**	7.94 %	30.03 %	23.61 %
I06	24.39 %	10.89 %	30.26 %	**34.46 %**
I07	**57.50 %**	2.67 %	15.68 %	24.14 %
I08	28.88 %	9.57 %	28.42 %	**33.14 %**
I09	30.12 %	8.38 %	**37.66 %**	23.84 %
I10	**78.01 %**	3.08 %	11.35 %	7.57 %
I11	15.62 %	8.94 %	**38.14 %**	37.30 %
I12	11.40 %	11.29 %	**41.57 %**	35.75 %

One of the most important advantages of the proposed prediction method is the ability to choose between the LOP stage and the HOP stage for each image block. This decision is taken based on RDO criteria, which allows the proposed HEVC-HOP to outperform HEVC-SS in all cases. From Table 5.3 it is possible to observe that despite the HOP stage of the proposed prediction method being used more frequently than the LOP stage, there is always a considerable part of the image that is encoded using only the LOP prediction mode. However, the fact that the HOP stage is used more often than the LOP stage alone indicates that, although the proposed method HEVC-HOP-P requires additional overhead for transmitting the prediction information, i.e., one flag (F_{HOP}) and four vectors ($\{\vec{v}_{HOP_n}\}$), it is very efficient in reducing the distortion between the current block and prediction block, therefore reducing the RD cost. An example of this can be seen in Figure 5.7 where a comparison between the generated prediction block using either the LOP method or the proposed HOP method is shown. In this example, the prediction block generated using the proposed HOP stage has a lower RD cost ($J_{HOP} = 5080$) than the prediction block using LOP ($J_{LOP} = 7851$), despite the extra bits necessary to convey to the decoder the GT parameters ($\{\vec{v}_{HOP_n}\} = \{(3;0), (4;0), (2;-1), (2;1)\}$).

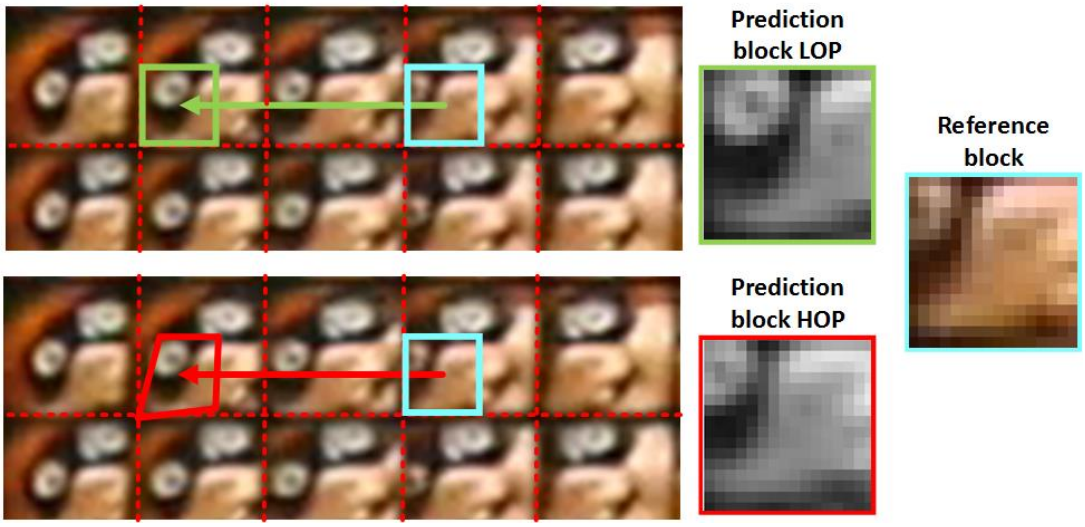


Figure 5.7. Comparison between the prediction block generated by LOP and HOP stages.

Results for different lenslet LF camera models

In general, the bitrate savings across the different codecs when compared to HEVC are higher when encoding LF images captured with cameras using a focused LF camera model. This is possible to see when comparing the average bitrate savings for the LF images captured with an unfocused camera model or the focused camera model in Table 5.1. This occurs because HEVC-SS and HEVC-HOP are based on matching prediction tools. In the focused images, the MIs are focused, therefore sharper than the unfocussed images. In sharper MI, more prominent features exist and therefore the block matching is more reliable [52]. Additionally, since the incident light in the camera's sensor in the unfocused case is focused at infinity, the disparity between MIs tends to be zero, which means that theoretically no perspective compensation can be matched. This can be justified by the noticeable lower relative prediction mode usage, shown in Table 5.3, for the proposed prediction method as well as SS-Skip for most LF images captured with unfocused camera models.

The proposed HOP model is also more suited to adapt to non-rectangular shape MIs, e.g., hexagonal and circular shape, when compared to LOP model based methods. This happens because the corners of the prediction blocks, when using the proposed HOP model, are flexible to adapt for different block shapes.

Comparison using the JPEG Pleno CTCs

To further evaluate the compression efficiency of the proposed HEVC-HOP-P, the experimental results were also performed using the 4DLF-MI data representation and the JPEG Pleno CTCs (see section 2.6.2). Since the LF images are encoded using the YUV 4:2:0 8 bit color format the adaptations shown in Figure 4.6, in Section 4.1.2 were applied. Table 5.4 shows the experimental results using the JPEG Pleno CTCs comparing HEVC, HEVC-SS (2 DoF) and HEVC-HOP-P (8 DoF).

Table 5.4. BD-PSNR-YUV and BD-Rate results using the JPEG Pleno CTCs comparing HEVC, HEVC-SS (2 DoF) and HEVC-HOP-P (8 DoF)

Image	HEVC-SS (2 DoF) vs HEVC		HEVC-HOP-P (8 DoF) vs HEVC-SS (2 DoF)	
	BD-PSNR-YUV	BD-RATE	BD-PSNR-YUV	BD-RATE
I01	2.65 dB	-60.17 %	0.13 dB	-4.39 %
I02	1.71 dB	-46.12 %	0.12 dB	-3.92 %
I03	1.36 dB	-37.36 %	0.08 dB	-2.68 %
I04	2.48 dB	-60.81 %	0.01 dB	-0.23 %
I05	2.32 dB	-62.33 %	0.08 dB	-3.55 %
I06	5.00 dB	-90.69 %	0.07 dB	-2.46 %
I07	4.67 dB	-73.34 %	0.17 dB	-5.88 %
I08	5.99 dB	-87.18 %	0.02 dB	-1.02 %
I09	2.87 dB	-62.05 %	0.17 dB	-5.53 %
I10	4.22 dB	-73.73 %	0.00 dB	-0.11 %
I11	4.32 dB	-83.37 %	0.33 dB	-14.87 %
I12	4.68 dB	-75.08 %	0.37 dB	-11.27 %
AVG.	3.52 dB	-67.69 %	0.13 dB	-4.66 %

From the results presented in Table 5.4 it is possible to observe that there is a coherence between the results for the BEPC and the JPEG Pleno CTCs using different LF data representations. The average bitrate savings achieved by HEVC-HOP-P relative to HEVC-SS are very similar for both cases. Nevertheless, since the images in 4DLF-MI are 11% larger than the LF images using the LL data representation, the inter-MI redundancy is even higher, therefore, solutions like HEVC-SS are even more efficient in this data representation when compared to HEVC, which does not exploit the inter-MI redundancy.

5.4.3. HOP training mode performance assessment

The experimental results for the above-mentioned test images using the BEPC comparing HEVC, HEVC-SS, HEVC-HOP-P and HEVC-HOP-*n*T using one, three and seven training directions are shown in Table 5.5. The values in bold correspond to the best achieved result for each specific test image.

Table 5.5. BD-PSNR-Y and BD-RATE results using BEPC comparing HEVC, HEVC-SS, HEVC-HOP and HEVC-HOP-nT, using one, three and seven training directions

Image	HEVC-SS vs HEVC		HEVC-HOP-P vs HEVC-SS		HEVC-HOP-1T vs HEVC-SS		HEVC-HOP-3T vs HEVC-SS		HEVC-HOP-7T vs HEVC-SS	
	BD-PSNR-Y	BD-RATE	BD-PSNR-Y	BD-RATE	BD-PSNR-Y	BD-RATE	BD-PSNR-Y	BD-RATE	BD-PSNR-Y	BD-RATE
PT0	0.90 dB	-14.64 %	**0.27 dB**	**-4.66 %**	0.07 dB	-1.20 %	0.12 dB	-2.18 %	0.17 dB	-3.06 %
PT150	1.44 dB	-19.02 %	**0.75 dB**	**-11.05 %**	0.28 dB	-4.12 %	0.47 dB	-6.98 %	0.56 dB	-8.30 %
DS	1.09 dB	-31.43 %	0.26 dB	-8.39 %	0.18 dB	-5.71 %	0.24 dB	-7.50 %	**0.29 dB**	**-9.14 %**
DC	1.05 dB	-29.25 %	**0.30 dB**	**-9.14 %**	0.17 dB	-5.41 %	0.24 dB	-7.43 %	0.30 dB	-9.09 %
Laura	2.26 dB	-30.35 %	0.27 dB	-4.78 %	0.15 dB	-2.65 %	0.34 dB	-6.11 %	**0.40 dB**	**-7.09 %**
Seagull	2.81 dB	-42.78 %	0.31 dB	-6.82 %	0.23 dB	-5.09 %	0.51 dB	-11.08 %	**0.59 dB**	**-12.57 %**
I01	0.81 dB	-18.50 %	**0.27 dB**	**-4.66 %**	0.07 dB	-1.20 %	0.12 dB	-2.18 %	0.17 dB	-3.06 %
I02	0.61 dB	-14.67 %	**0.75 dB**	**-11.05 %**	0.28 dB	-4.12 %	0.47 dB	-6.98 %	0.56 dB	-8.30 %
I03	0.17 dB	-4.10 %	0.26 dB	-8.39 %	0.18 dB	-5.71 %	0.24 dB	-7.50 %	**0.29 dB**	**-9.14 %**
I04	0.25 dB	-6.31 %	**0.30 dB**	**-9.14 %**	0.17 dB	-5.41 %	0.24 dB	-7.43 %	0.30 dB	-9.09 %
I05	0.89 dB	-28.73 %	0.27 dB	-4.78 %	0.15 dB	-2.65 %	0.34 dB	-6.11 %	**0.40 dB**	**-7.09 %**
I06	1.43 dB	-45.35 %	0.31 dB	-6.82 %	0.23 dB	-5.09 %	0.51 dB	-11.08 %	**0.59 dB**	**-12.57 %**
I07	0.48 dB	-13.73 %	**0.13 dB**	**-3.45 %**	0.01 dB	-0.18 %	0.03 dB	-1.01 %	0.06 dB	-1.58 %
I08	0.66 dB	-22.95 %	**0.11 dB**	**-2.94 %**	0.00 dB	-0.06 %	0.02 dB	-0.58 %	0.06 dB	-1.52 %
I09	1.49 dB	-30.72 %	**0.03 dB**	**-0.77 %**	0.00 dB	-0.01 %	0.01 dB	-0.19 %	0.02 dB	-0.37 %
I10	0.22 dB	-8.10 %	**0.02 dB**	**-0.57 %**	0.00 dB	0.00 %	0.00 dB	-0.04 %	0.01 dB	-0.15 %
I11	1.49 dB	-42.84 %	**0.14 dB**	**-5.30 %**	0.00 dB	-0.46 %	0.04 dB	-1.59 %	0.06 dB	-2.48 %
I12	1.42 dB	-41.35 %	**0.06 dB**	**-4.46 %**	0.00 dB	0.99 %	0.00 dB	-1.14 %	0.02 dB	-1.99 %
AVG. FOC	1.59 dB	-27.91 %	0.36 dB	-7.47 %	0.18 dB	-4.03 %	0.32 dB	-6.88 %	**0.39 dB**	**-8.21 %**
AVG. UNF	0.83 dB	-23.11 %	**0.14 dB**	**-5.06 %**	0.02 dB	-0.65 %	0.04 dB	-1.34 %	0.06 dB	-2.14 %
AVG. ALL	1.08 dB	-24.71 %	**0.21 dB**	**-5.86 %**	0.07 dB	-1.78 %	0.13 dB	-3.19 %	0.17 dB	-4.16 %

When comparing HEVC with HEVC-SS, which is limited to only two DoF, it is possible to see that HEVC-SS is able to outperform HEVC with average bitrate savings of 27.91% and 23.11% for LF images using FOC and UNF camera models, respectively. However, when comparing HEVC-HOP, which supports up to eight DoF, with HEVC-SS, additional average bitrate savings of 7.47% and 5.06% are achieved for LF images using FOC and UNF camera models, respectively.

When comparing the results for the proposed HEVC-HOP-nT it is possible to see that the bitrate savings increase for every image when the number of training directions is increased. For seven training directions, i.e., HEVC-HOP-7T, the achieved average bitrate savings, for the LF images captured by a camera using a FOC model when compared to HEVC-SS, is 8.21% (up to 12.57%). Additionally, when compared to HEVC the average bitrate savings is 33.55% (up to 50.03%). In this case, HEVC-HOP-7T is able to outperform HEVC-HOP. However, when encoding images from the EPFL LF dataset using HEVC-HOP, the average bitrate savings relatively to HEVC-SS is 5.06%, where when the proposed HEVC-HOP-7T is used, average bitrate savings of only 2.14% are achieved. Concluding that for LF images captured with a UNF camera model, HEVC-HOP is more efficient than the proposed HEVC-HOP-7T. The number of training directions can be further increased, however the bitrate savings gains for a higher number of training directions is residual and the encoder computational complexity is vastly increased.

Comparison using the JPEG Pleno CTCs

To further evaluate the compression efficiency of the proposed HEVC-HOP-nT, the experimental results were also performed using the 4DLF-MI data representation and the JPEG Pleno CTCs. In this case only the EPFL LF dataset is used due to the metadata requirement to convert from the LL image to the 4D LF data representation. Since the LF images are encoded using the YUV 4:2:0 8 bit color format the adaptations shown in Figure 4.6, in Section 4.1.2 were applied Table 5.6 shows the experimental results using the JPEG Pleno CTCs comparing HEVC, HEVC-SS, HEVC-HOP-P and HEVC-HOP-7T. The values in bold correspond to the best achieved result for each specific test image.

The results presented in Table 5.6 further validate the assumption that although HEVC-HOP-7T is able to be more efficient than a LOP solution such as HEVC-SS, it is not as efficient as HEVC-HOP-P for LF images captured with a UNF camera model. Regardless, the average bitrate savings increased by about 1% over HEVC-SS, when performing the tests with the 4DLF-MI representation using the JPEG Pleno CTCs (3.02%), instead of using the LL representation with the BEPC (2.14%).

Table 5.6. BD-PSNR-YUV and BD-Rate results using the JPEG Pleno CTCs comparing HEVC, HEVC-SS, HEVC-HOP-P and HEVC-HOP-7T

Image	HEVC-SS vs HEVC		HEVC-HOP-P vs HEVC-SS		HEVC-HOP-7T vs HEVC-SS	
	BD-PSNR-YUV	BD-RATE	BD-PSNR-YUV	BD-RATE	BD-PSNR-YUV	BD-RATE
I01	2.65 dB	-60.17 %	**0.13 dB**	**-4.39 %**	0.08 dB	-2.53 %
I02	1.71 dB	-46.12 %	**0.12 dB**	**-3.92 %**	0.07 dB	-2.27 %
I03	1.36 dB	-37.36 %	**0.08 dB**	**-2.68 %**	0.03 dB	-0.85 %
I04	2.48 dB	-60.81 %	**0.01 dB**	**-0.23 %**	0.00 dB	0.14 %
I05	2.32 dB	-62.33 %	**0.08 dB**	**-3.55 %**	0.07 dB	-2.81 %
I06	5.00 dB	-90.69 %	0.07 dB	-2.46 %	**0.10 dB**	**-4.56 %**
I07	4.67 dB	-73.34 %	**0.17 dB**	**-5.88 %**	0.08 dB	-2.78 %
I08	5.99 dB	-87.18 %	**0.02 dB**	**-1.02 %**	0.00 dB	0.18 %
I09	2.87 dB	-62.05 %	**0.17 dB**	**-5.53 %**	0.10 dB	-3.32 %
I10	4.22 dB	-73.73 %	0.00 dB	-0.11 %	**0.03 dB**	**-0.92 %**
I11	4.32 dB	-83.37 %	**0.33 dB**	**-14.87 %**	0.23 dB	-10.37 %
I12	4.68 dB	-75.08 %	**0.37 dB**	**-11.27 %**	0.19 dB	-6.09 %
AVG.	3.52 dB	-67.69 %	**0.13 dB**	**-4.66 %**	0.08 dB	-3.02%

5.5. Final remarks

In this chapter, a HOP mode and HOP training for LF image coding was proposed, using GTs of up to eight DoF. The proposed HOP model is a two-stage block-wise approach that is able to achieve RD efficiency gains, relative to a LOP state-of-the-art solution for LF image coding and HEVC. These gains occur, regardless of the LF camera model, MI representation, LF image resolution and microlens array type. Experimental results using the BEPC show that the proposed HOP mode achieves an average bitrate savings of 5.86% and 28.81%, when compared to HEVC-SS and HEVC, respectively, across different types of LF images, when using the Projective GT. It is also possible to conclude that, the GTs with eight DoF, namely Projective and Bilinear, are generally more efficient than Affine GT in a RD sense. Additionally, the proposed HOP training is able to outperform HEVC-SS and HEVC by an average bitrate savings of 4.16% and 27.37%, respectively. It was observed that the HOP training is able to outperform the HOP mode when encoding images captured with a FOC LF camera model, however, the opposite was observed for UNF LF camera models.

The same proposed solutions were also tested when encoding the EPFL LF dataset, using JPEG Pleno CTCs. In this case the average bitrate savings achieved for the HOP mode is 4.81% and

69.86% when over HEVC-SS and HEVC, respectively. When comparing the HOP training using the same evaluation metrics, the average bitrate savings achieved are 3.02% and 68.95% when compared to HEVC-SS and HEVC, respectively.

Chapter 6. Light field image coding based on hybrid data representation

Most techniques reviewed in Chapter 3 either rely on exclusively exploiting the redundancy of MIs or SAIs which limits the amount of redundancy that the LF image coding is able to exploit, thus limiting its overall efficiency. This chapter proposes a hybrid LF data representation (HR), i.e., one which uses both the MI and SAI representations, enabling a more exhaustive exploitation of the inherent LF redundancy and improving the LF coding efficiency. The use of the hybrid approach enables the use of four main types of redundancy: (i) intra spatial redundancy of each SAI, (ii) inter-view redundancy between SAIs, (iii) intra-MI redundancy within each MI, and (iv) inter-MI between neighboring MIs.

The efficiency of the proposed HR is demonstrated by incorporating this new paradigm in a standard HEVC PVS codec [64]. In this codec, spatial redundancy of each SAI can be exploited by standard intra coding tools. The inter-view redundancy between SAIs and the intra and inter-MI redundancies are exploited by using a new hybrid reference picture list (HRPL). The proposed HRPL, allows the already encoded LF information to be stored simultaneously in a SAI- and a MI-based manner, which allows all types of redundancy to be exploited, by applying different prediction modes.

The proposed codec uses an optimized set of pixelwise prediction methods, evaluated amongst state-of-the-art methods such as: DC predictor, median edge detector (MED) [110], gradient adjusted predictor (GAP) [111] and accurate gradient selective prediction (AGSP) [112]. Additionally, new prediction methods based on least squares prediction (LSP) [113] are proposed, to further improve the coding efficiency. Experimental results show that the proposed

codec using this HR is able to outperform state-of-the-art LF image codecs that rely exclusively on either MI- or SAI-based representations.

This chapter is organized as follows: Section 6.1 presents the proposed hybrid LF data representation; Section 6.2 describes the new intra-MI prediction modes proposed in this Chapter; Section 6.3 presents the new inter-MI prediction modes proposed in this chapter, Section 6.4 evaluates the performance of the proposed LF coding solution against relevant state-of-the-art solutions; and finally, Section 6.5 concludes this chapter with some final remarks.

6.1. Proposed hybrid LF data representation

The HR proposed in this chapter uses a combination of the 4DLF-PVS and 4DLF-MI representations. This LF data representation takes advantage of the seamless and reversible conversion between 4DLF-PVS and 4DLF-MI representations (see Section 2.2). This hybrid representation enables the use of more prediction paradigms (modes) in the compression of LF data. Figure 6.1 shows the coding diagram of a video codec which uses the hybrid representation in a HEVC-like encoder, which employs the proposed hybrid data representation.

From Figure 6.1 it is possible to see the use of two decoded picture buffers: (i) the SAI decoded picture buffer, which is the standard HEVC decoded picture buffer, and (ii) the new MI decoded picture buffer that stores the full LF image using the 4DLF-MI representation, which is gradually generated from the decoded SAIs. The combination of both decoded picture buffers is referred to as HRPL.

The use of the HRPL enables four types of estimation/prediction blocks that exploit different types of redundancy available in each data representation model:

- **Intra-SAI prediction**: Corresponds to the intra-picture prediction modes, i.e., DC, Planar, and Directional modes in HEVC, which are used to exploit the spatial redundancy of SAIs;
- **Inter-SAI prediction**: Corresponds to the inter-picture prediction modes, i.e., Motion Estimation, Merge/Skip modes in HEVC, which are used to exploit the inter-view redundancy of SAIs. These prediction modes make use of the SAI decoded picture buffer;

- **Intra-MI prediction**: Corresponds to new prediction modes that exploit the intra-MI redundancy, by using the MI decoded picture buffer (see Section 6.2);
- **Inter-MI prediction**: Corresponds to new prediction modes that exploit the inter-MI redundancy, by using the MI decoded picture buffer (see Section 6.3).

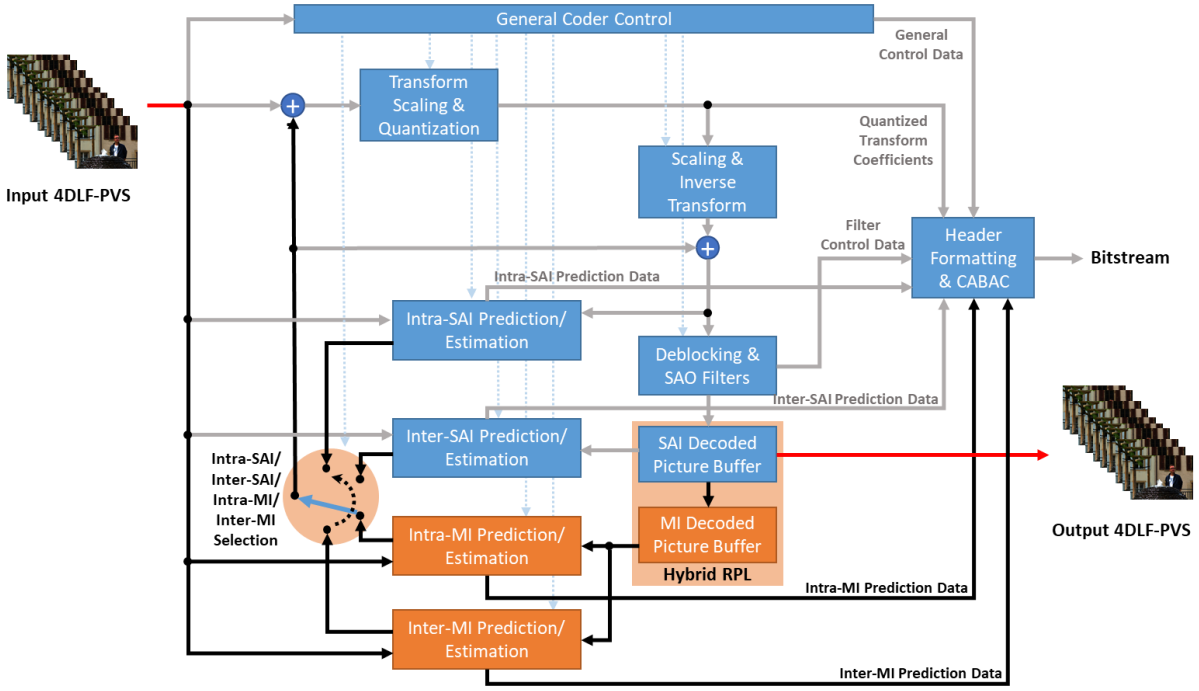


Figure 6.1. Proposed LF coding architecture using the hybrid data representation. The highlighted portions in orange correspond to the contributions of this work.

6.1.1. Generation of the MI decoded picture buffer

As illustrated in Figure 6.1, in the proposed LF coding solution, both input and output data use the 4DLF-PVS data representation. Therefore, the full LF image using the 4DLF-MI data representation must be generated from the decoded SAIs, by using the correspondence between the 4DLF-PVS and 4DLF-MI representations. The pixel position on the 4DLF-MI image, (i, j), can be defined as a function of the 4DLF-PVS pixel position:

$$\begin{pmatrix} i \\ j \end{pmatrix}_{4DLF-MI} = \begin{pmatrix} h(f_n) + xM_w \\ v(f_n) + yM_h \end{pmatrix}_{4DLF-PVS}, \tag{6.1}$$

where, M_w and M_h correspond to the MI width and height, respectively and f_n is the PVS frame number. As mentioned before, the (x, y) coordinates index the pixel position within each SAI and (h, v) coordinates index the SAI position. Since the SAIs are organized in a PVS the (h, v)

coordinates depend on the frame number f_n. Figure 6.2 shows how the 4DLF-MI image is generated from the nine decoded SAIs when a spiral scan is used. Note that the dashed red arrow shows the scanning order being applied on the 4DLF-PVS representation and the consequent scanning order from the 4DLF-MI representation point of view. When applying (6.1) to the example in Figure 6.2 the pixel at the $P_{PVS}(x, y, f_n)$ coordinate in the 4DLF-PVS representation is copied to the $P_{MI}(i, j)$ coordinate on the 4DLF-MI representation. This allows the 4DLF-MI decoded picture buffer to be gradually filled after encoding each full SAI.

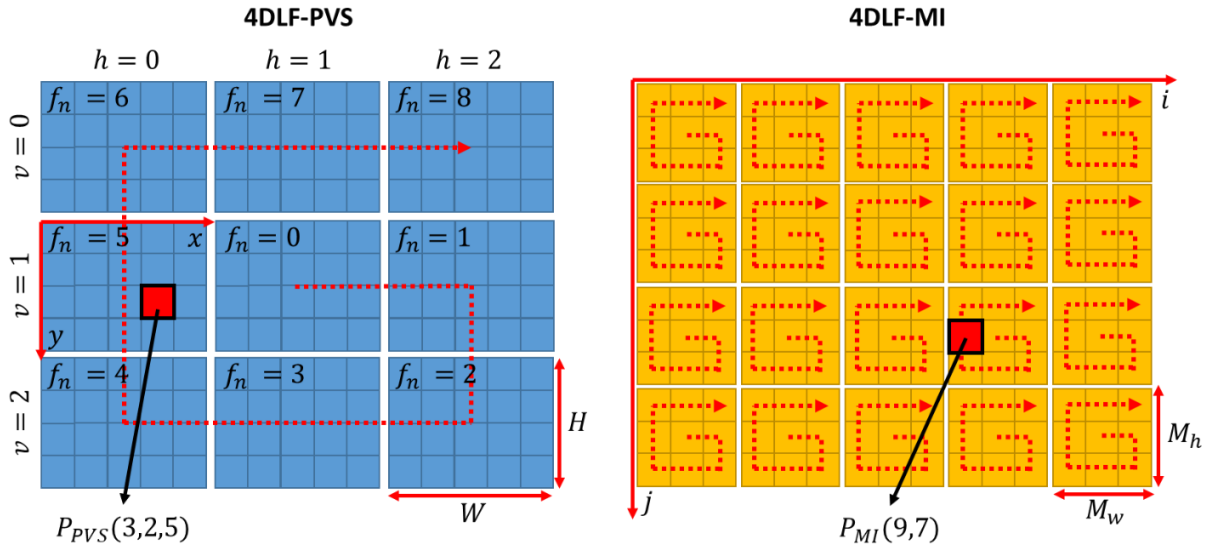


Figure 6.2. Pixel correspondence between 4DLF-PVS and 4DLF-MI data representations.

Since the conversion to the 4DLF-MI representation is performed progressively the 4DLF-MI decoded picture buffer is going to resemble a sparse LF image. Figure 6.3 shows the conversion of the first 2×2 block when encoding the sixth SAI of the 4DLF-PVS from the example shown before in Figure 6.2. As it is possible to see from the example in Figure 6.3, the reference 2×2 block in the 4DLF-PVS representation, leads to four individual pixels in four different MIs, in the 4DLF-MI representation. Because of this characteristic, the intra-MI and inter-MI prediction modes are applied pixelwise, instead of blockwise, as in the 4DLF-PVS representation case. In the example of Figure 6.4, each one of the four individual pixels is predicted using either the causal area in the same MI (intra-MI) or the causal area in neighboring MIs (inter-MI). After pixelwise pixel prediction, the predicted pixels are mapped back to the 4DLF-PVS representation positions forming a prediction block for the reference 2×2 block.

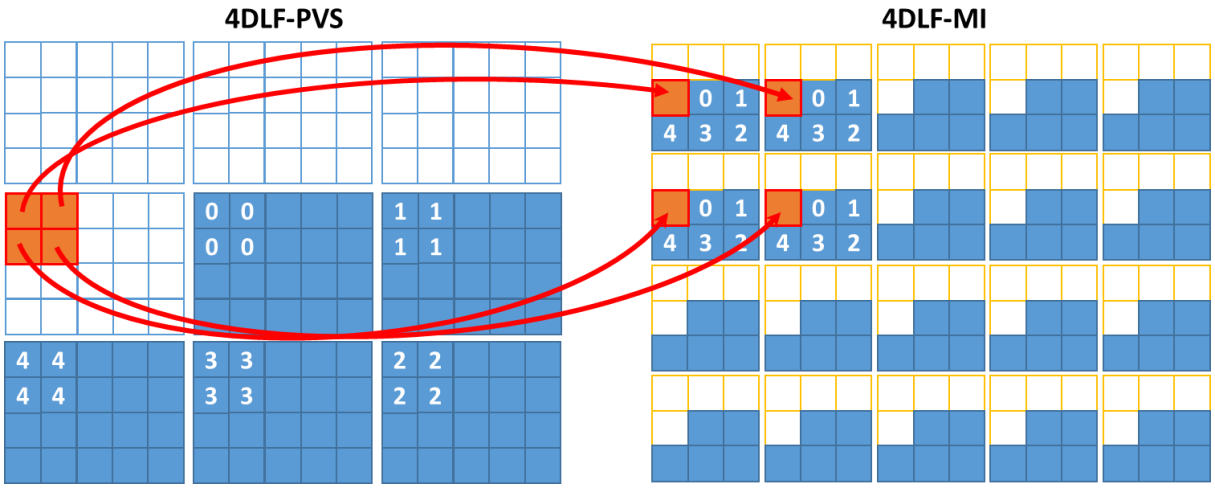


Figure 6.3. Converting a reference 2×2 block in the 4DLF-PVS representation to the 4DLF-MI representation.

6.1.2. Selection of the 'best' prediction mode

Several prediction modes are going to be available to exploit each type of redundancy. The HEVC encoder decides between both intra and inter prediction modes, i.e., intra-SAI and inter-SAI prediction modes, respectively, by generating a prediction block for each prediction mode and then comparing the predicted block with the original block. The prediction mode that minimizes a Lagrangian RD cost function, given by $J = D + \lambda R$, is selected. In the proposed coding approach, the same process is extended to the proposed prediction modes to be used in the 4DLF-MI data representation. The prediction modes in both representation domains are going to be used to also generate prediction blocks, which will then compete, in terms of RD cost, with the ones generated by the standard HEVC prediction modes.

6.1.3. Prediction mode signaling

Since in the proposed coding architecture four different types of redundancies can be exploited with the different prediction modes, it is also necessary to efficiently signal the usage of such prediction modes. The standard intra and inter modes, i.e., intra-SAI and inter-SAI, from HEVC are signaled as in the HEVC standard. The proposed intra-MI and inter-MI prediction modes also use the same signaling rational, from the 35 possible intra-SAI prediction modes, eight directional mode indexes are allocated for intra-MI modes and inter-MI modes. The substituted modes are the suggested ones in [48] which include intra directions that are not used as often as

the others. Table 6.1 shows the list of used modes (name/number) for each of the four prediction types.

Table 6.1. List of mode name/number for the proposed coding structure

Prediction type	Mode name/index
intra-SAI	0 (Planar), 1 (DC), 2, 4, 5, 6, 8, 9, 10, 12, 13, 14, 16, 17, 18, 20, 21, 22, 24, 25, 26, 28, 29, 30, 32, 33, 34
Inter-SAI	Skip, Merge, Motion Compensation
intra-MI	3, 7, 11, 15
Inter-MI	19, 23, 27, 31

6.2. Intra-MI prediction

The intra-MI prediction modes are responsible to exploit intra-MI redundancy which is present in each individual MI. In order to exploit this type of redundancy several modes are proposed, such as DC, MED [110], GAP [111], and AGSP [112]. These modes, with the exception of AGSP, are used in popular image coding approaches, such as HEVC, JPEG-LS [114] and CALIC [111], respectively. AGSP was selected because it is able to outperform MED and GAP when encoding natural images. However, such prediction modes cannot be used directly in the proposed codec. An adaptation of the prediction area is necessary because the causal area may differ, depending on the scanning strategy adopted. The following sections describe the proposed pixel prediction modes, including the necessary adaptations to the spiral scanning strategy.

6.2.1. Spiral scanning predictor adaptations

In the proposed hybrid representation, a clockwise spiral PVS scan is performed, causing the causal area to grow differently from a raster scan. When using a raster scan, as in the case of the original DC, MED, GAP and AGSP predictors, the causal area is always on the top and left of the current pixel. The predicted pixel generated by each of these predictors is a combination of part or all of the available pixels in the causal area. For example, the predicted pixel that is generated by the DC predictor is an average, i.e., a linear combination, of the surrounding pixels. The set of pixels that are selected within the causal area of pixels are hereafter referred to as

pixel support. Differently from the raster scan, for the clockwise spiral scan, the causal area of each new pixel is not always available in the upper-left area relative to the new pixel. Thus, the pixel support is dynamically adapted to maximize the number of available pixels for prediction, which is illustrated in Figure 6.4 for Frame 33 of a PVS, where each individual predictor is used to predict the orange pixel.

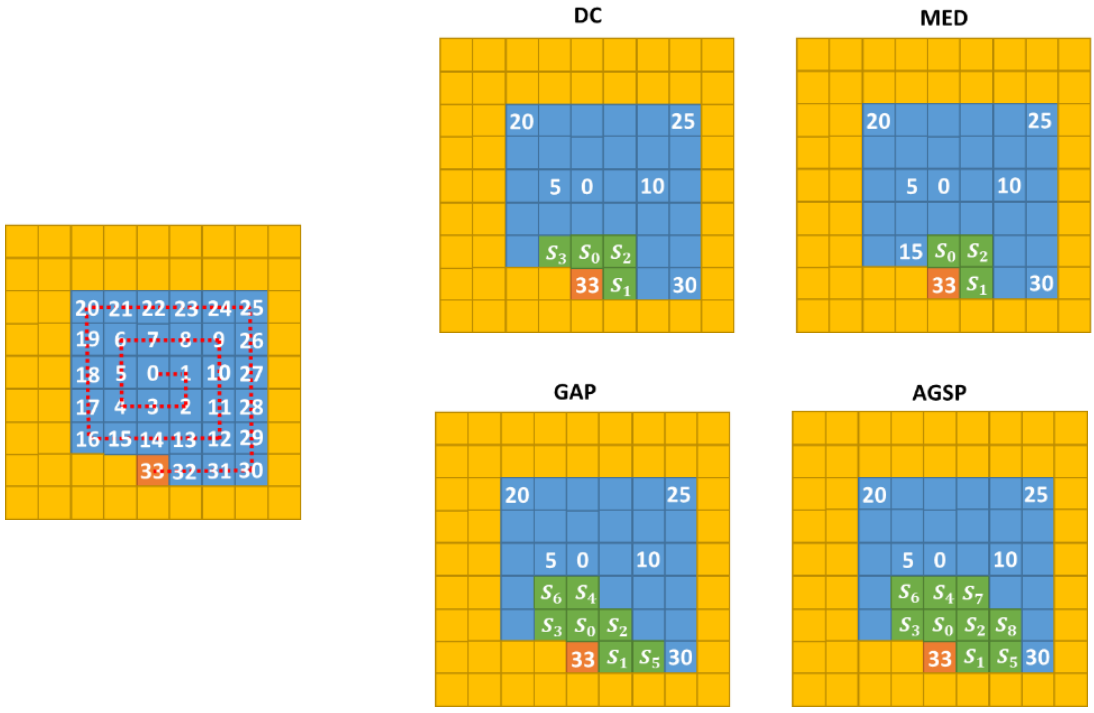


Figure 6.4. Pixel support for the pixel predictors DC, MED, GAP and AGSP, when using a clockwise spiral scan.

The spiral scanning is divided into four phases, i.e., Right, Down, Left and Up, named according to the direction of the spiral scan. Figure 6.4 shows a pixel prediction, the causal area and the pixel support, when the scan direction is left (i.e., Left phase). For the following phases (Up, Right and Down) the same pixel prediction structure is used, but rotated relatively to the Left phase as follows:

- **Up phase:** 90º;
- **Right phase:** 180º;
- **Down phase:** 270º.

At few positions of the scan, one or more pixels of the pixel support area may not exist. When some predictor pixel value is not available, due to being part of the non-causal area, it is copied

from a neighboring location. The filling pattern for the unavailable pixels is shown in Figure 6.5 with red arrows.

Figure 6.5. Generic pixel support and filling pattern.

6.2.2. DC prediction mode

The DC prediction mode consists in applying an average of the available pixels in a 3×3 template centered on the current pixel. In the example shown in Figure 6.4 the predictor is an average of the values of pixels $S_0, \cdots, S_3$. If N is the number of available pixels, the prediction value $\hat{P}$ is generically calculated by (6.2):

$$\hat{P} = \frac{\sum_0^{N-1} S_n}{N}, \tag{6.2}$$

The number of available pixels varies between one and four. Notice that the decoder has access to the same predictor values, since a causal prediction is used.

6.2.3. MED prediction mode

The MED [110] prediction mode consists of a three-pixel template, as shown in Figure 6.4. The prediction value is calculated by (6.3):

$$\hat{P} = \begin{cases} \min(S_1, S_0), if\ S_2 \geq \max(S_1, S_0) \\ \max(S_1, S_0), if\ S_2 \leq \min(S_1, S_0) \\ S_1 + S_0 - S_2, \quad otherwise \end{cases}. \tag{6.3}$$

6.2.4. GAP prediction mode

The GAP [111] prediction mode consists of a seven-pixel template, as shown in Figure 6.4. Firstly, the vertical (g_v) and horizontal (g_h) gradients are estimated using (6.4):

$$\begin{aligned} g_h &= |S_1 - S_5| + |S_0 - S_2| + |S_0 - S_3| \\ g_v &= |S_1 - S_2| + |S_0 - S_4| + |S_3 - S_6|, \end{aligned} \tag{6.4}$$

Secondly, depending on the values of g_v and g_h, GAP will recognize weak, regular and sharp vertical (Ver) and horizontal (Hor) axis as well as smooth edges. The prediction value $\hat{P}$ is determined by the thresholds [111] and equations shown in Figure 6.4.

Table 6.2. Prediction value calculation based on the vertical and horizontal gradients

Edge	Threshold ($g_v - g_h$)	Prediction ($\hat{P}$)
Sharp Hor	> 80	S_1
Sharp Ver	< -80	S_0
Regular Hor	> 32	$\frac{S_1 + S_0}{4} + \frac{S_3 - S_2}{8} + \frac{S_1}{2}$
Regular Ver	< -32	$\frac{S_1 + S_0}{4} + \frac{S_3 - S_2}{8} + \frac{S_0}{2}$
Weak Hor	> 8	$\frac{3(S_1 + S_0)}{8} + \frac{3(S_3 - S_2)}{16} + \frac{S_1}{4}$
Weak Ver	< -8	$\frac{3(S_1 + S_0)}{8} + \frac{3(S_3 - S_2)}{16} + \frac{S_0}{4}$
Smooth	otherwise	$\frac{S_1 + S_0}{2} + \frac{S_3 - S_2}{4}$

6.2.5. AGSP prediction mode

The AGSP [112] prediction mode uses a nine-pixel predictor as, shown in Figure 6.4. AGSP is able to determine horizontal, vertical and diagonal edges. In order to determine the direction of the edge, 4 gradients are calculated in (6.5), corresponding to the horizontal (g_h), vertical (g_v), 45º diagonal (g_{45}) and -45º diagonal (g_{-45}):

$$\begin{aligned} g_h &= (2|S_1 - S_5| + 2|S_0 - S_2| + 2|S_0 - S_3| + |S_4 - S_7| + |S_4 - S_6| + |S_2 - S_8|)/9 + 1 \\ g_v &= (2|S_1 - S_2| + 2|S_0 - S_4| + |S_3 - S_6| + |S_5 - S_8| + |S_2 - S_7|)/7 + 1 \\ g_{45} &= (2|S_1 - S_0| + 2|S_0 - S_6| + |S_5 - S_2| + |S_2 - S_4|)/6 + 1 \\ g_{-45} &= (2|S_1 - S_8| + 2|S_0 - S_7| + |S_3 - S_4|)/5 + 1 \end{aligned} \quad (6.5)$$

After calculating the four gradients, the two lowest ones are selected as g_{min} and g_{min2}. Additionally, the causal pixels P_{min} and P_{min2} that correspond to the direction of each of the selected gradients, i.e., $g_{\min}$ and $g_{\min 2}$, are selected. The correspondent causal pixel for the gradients g_v, g_h, g_{45} and g_{-45}, is S_0, S_1, S_2 and S_3, respectively. For example, if $g_{\min} = g_v$

and $g_{\min 2} = g_{45}$, then $P_{min} = S_0$ and $P_{min2} = S_2$. The final prediction value is calculated in (6.6):

$$\hat{P} = \frac{g_{min}\ P_{min2} + g_{min2}\ P_{min}}{g_{min} + g_{min2}} \tag{6.6}$$

6.3. Inter-MI prediction

The inter-MI prediction modes are responsible to exploit the inter-MI redundancy which is also known, as non-local spatial redundancy. The similarities between the neighboring MIs in the LF image using the 4DLF-MI representation can be exploited in several ways as it was discussed in Chapter 3. However, most approaches proposed in the literature are block-based instead of pixel-based. This work describes the use of an LSP-based prediction mode which can be applied in a pixelwise manner, in order to exploit the inter-MI redundancy.

6.3.1. LSP- based prediction mode

LSP is a prediction tool that adaptively estimates optimal linear coefficients using least squares training. Thus, the main advantage of the LSP-based prediction mode, when compared with the previously presented predictors, is its ability to dynamically adapt to the available causal area and determine the best prediction direction depending on the causal area [113]. The least squares training step based on a least squares minimization problem is defined as:

$$\min_{\boldsymbol{a}}(\|\boldsymbol{y} - \boldsymbol{C}\boldsymbol{a}\|_2^2) \tag{6.7}$$

where $\boldsymbol{a} = [a_0, \dots, a_{M-1}]^T$, a $M \times 1$ column vector, corresponds to the linear coefficients to estimate. The closed-form solution for (6.7) is given by:

$$\boldsymbol{a} = (\boldsymbol{C}^T\boldsymbol{C})^{-1}(\boldsymbol{C}^T\boldsymbol{y}). \tag{6.8}$$

The matrix $\boldsymbol{C}$ is a $T \times M$ matrix, where M is the order of the LSP pixel support, i.e., the number of pixels that compose the pixel support, and T is the number of causal neighbors used for training, which can include the causal area in the current MI or several neighboring MIs. The vector $\boldsymbol{y}$ is a $T \times 1$ column vector, i.e., $\boldsymbol{y} = [y_0, \dots, y_{T-1}]^T$. The value T is calculated as (6.9):

$$T = (f_n - 1) \times M_T, \tag{6.9}$$

where f_n is the frame number of the PVS and M_T is the number of MIs used for training. Note that if $M_T = 1$ this mode can be considered intra-MI, because the training step only uses pixels from the current MI. If $M_T > 1$ then, this prediction mode is considered an inter-MI prediction mode. Additionally, the training size increases with the frame number, because more pixels are available for training.

6.3.2. Adaptive pixel support and training

In order to perform the LSP training, the pixel support, i.e., the pixels used for prediction after the training step is performed, for the predictor needs to be determined as well as the matrix $\boldsymbol{C}$ and the vector $\boldsymbol{y}$ need to be constructed. In this approach, since the causal area grows in a spiral, the pixel support needs to be adapted accordingly, by selecting the M available pixels closest to the current pixel. The distance between the current pixel, I_c, and the causal pixel, I_n, is determined by the Manhattan distance in (6.10):

$$d(I_c, I_n) = |I_{c_x} - I_{n_x}| + |I_{c_y} - I_{n_y}|. \tag{6.10}$$

For example, an adaptive pixel support generation, by minimizing the Manhattan distance, for $M = 5$ and $M = 9$, is shown in Figure 6.6; the numbers displayed in the neighboring pixels (blue pixels) correspond to the Manhattan distance of using as an example the pixel 33 shown in Figure 6.4, $T = 32 \times M_T$, because $F_n = 33$. The neighboring pixels that have the same Manhattan distance are selected by using a raster scan. High values of M_T may heavily increase the computational complexity, therefore, in this Thesis, only values between one and nine have been considered. In this case, each pixel relative to pixel 33. As illustrated in Figure 6.7, vector $\boldsymbol{y}$ comprises every single causal pixel inside the MIs that contain the training area, with the exception of the current pixel. The matrix $\boldsymbol{C}$ is composed by the pixel support determined in the previous step, centered on each neighboring pixel included in vector $\boldsymbol{y}$. Using the example shown in Figure 6.7, if using a fifth order pixel support, i.e., LSP using $M = 5$, this support is centered on pixel 0 and, therefore, the first row of the matrix $\boldsymbol{C}$ is $\boldsymbol{C_0} = [I_7, I_1, I_{22}, I_6, I_8]$ and $y_0 = I_0$.

Since the matrix $\boldsymbol{C}$ and vector $\boldsymbol{y}$ are already determined then (6.8) can be solved. After solving (6.8), vector $\boldsymbol{a}$ is used to estimate a prediction value $\hat{P}$ for the current pixel as:

$$\hat{P} = \sum_{0}^{M-1} S_n \times a_n, \tag{6.11}$$

where S_n are the M pixels that compose the pixel support.

Figure 6.6. Example of an adaptive pixel support generation by minimizing the Manhattan distance (No pixel support, M=5 and M=9).

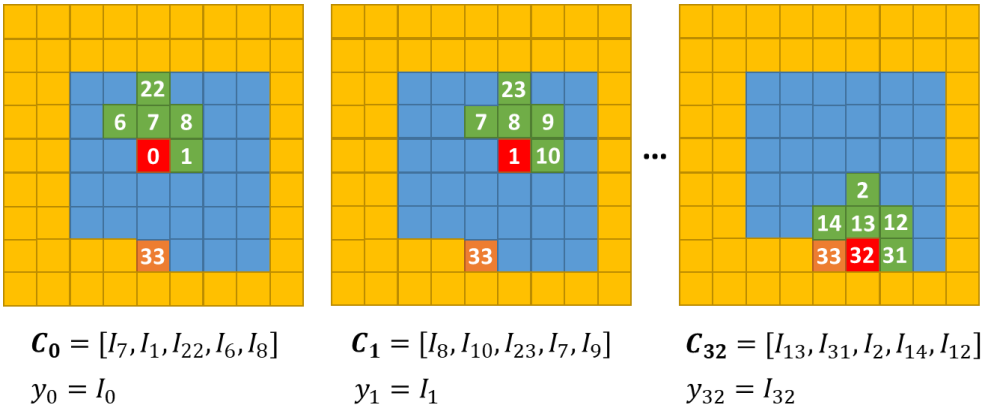


Figure 6.7. Composition of matrix C and vector y for the current MI necessary for the training step of LSP.

6.4. Experimental results

In this section the performance of the proposed LF coding solution based on a hybrid LF data representation is evaluated against the most relevant state-of-the-art coding solutions. First, the test conditions, including the processing chain for objective quality assessment, is explained. Then, experimental results comparing the RD performance of the proposed codec are presented and discussed. A statistical analysis of the prediction mode usage as well as the performance for different encoder configurations are evaluated and discussed.

6.4.1. Test conditions

The experimental tests for all coding solutions presented in this section adopted the JPEG Pleno CTCs [35]. The EPFL LF dataset (see Appendix A.2), comprised of twelve LF images acquired using a Lytro Illum camera, was used [22] to evaluate the several benchmarks. The state-of-the

art LF codecs that were used as benchmarks, include: HEVC-PVS [64], WaSP [42], MuLE [41], HEVC-SS [50], HEVC-HOP [11] and HEVC-LLE [48]. The proposed codec, based on a hybrid LF representation, is referred to as HEVC-HR. The LF input resolution, LF data representation and the supported color format, are presented in Table 6.3.

Table 6.3. Benchmark coding solutions

Codec	Input Resolution	Color format	LF Data Representation
HEVC-PVS [64]	625×434 (169 SAIs)	YUV 4:4:4 10 bit	SAI
WaSP [42]	625×434 (169 SAIs)	RGB 4:4:4 10 bit	SAI
MuLE [41]	625×434 (169 SAIs)	RGB 4:4:4 10 bit-	4D LF
HEVC-SS [50]	8125×5642 (13×13 MIs)	YUV 4:2:0 8 bit	MI
HEVC-HOP [11]	8125×5642 (13×13 MIs)	YUV 4:2:0 8 bit	MI
HEVC-LLE [48]	8125×5642 (13×13 MIs)	YUV 4:2:0 8 bit	MI
HEVC-HR	**625×434 (169 SAIs)**	**YUV 4:4:4 10 bit**	**Hybrid (SAI and MI)**

As mentioned before, the output color format for objective comparison of all benchmarks is YUV 4:4:4 10 bit. However some codec implementations only support the YUV 4:2:0 8 bit color format. This is the case for the HEVC-SS, HEVC-HOP and HEVC-LLE codecs. For these codecs, a pre-processing step is applied at the encoder, to generate the YUV 4:2:0 8 bit input color format, and a post-processing step is performed at the decoder to generate the YUV 4:4:4 10 bit output color format. In order to evaluate the achieved objective quality the processing chains shown in Figure 4.4 and Figure 4.6, (seen in Section 4.1.2) are used for the codecs that support YUV 4:4:4 10 bit and YUV 4:2:0 8 bit, respectively.

Table 6.4 shows the list of tested codecs including the corresponding configurations. The different QP and λ values selected allow the use of a common bitrate range for every tested codec, enabling a direct comparison through the BD metrics. The HEVC 4DLF-PVS based codecs use the low delay with B slices configuration and the 4DLF-MI based codecs use the intra main configuration.

Table 6.4. List of tested codecs and corresponding configurations

Codec	Configuration
HEVC-HR HEVC-PVS	$QP = [17,22,27,32,37,42]$
HEVC-SS HEVC-HOP HEVC-LLE	$QP = [22,27,32,37,42,47]$
MuLE	$\lambda = [270,3880,30000,310000,4600000]$
WaSP	$Target\ bpp = [0.001,0.005,0.02,0.1,0.75]$

6.4.2. Performance assessment

The proposed HEVC-HR was tested in three different phases. In the first phase, each intra-MI prediction mode was individually tested, i.e., DC, MED, GAP and AGSP. In the second phase, each inter-MI prediction mode was tested, i.e., several configurations in terms of LSP order and training area of the proposed LSP prediction mode were tested. In the final phase the prediction modes presenting a higher tradeoff between coding efficiency and computational complexity were selected to be part of HEVC-HR. The experimental results in Table 6.5 and Table 6.6 show the BD-PSNR-YUV and BD-RATE averaged across the twelve EPFL LF images, comparing the HEVC-PVS with the HEVC-HR using only one prediction mode.

Intra-MI prediction modes evaluation

Table 6.5 presents the results of each individual intra-MI prediction mode described in section 6.2. As can be observed, the prediction mode with highest bitrate savings (12.87%) is AGSP. From the experimental results it can be inferred that increasing the order of the prediction increases the prediction accuracy and, consequently, improves the coding efficiency. The only exception is the MED prediction mode, which achieves lower bitrate savings when compared to the DC prediction mode, which has an order value of one to four, depending on how many support pixels are available. Overall, it is possible to observe that the proposed intra-MI prediction modes improve the LF image coding efficiency. Since their low computational complexity, especially when compared to LSP-based prediction modes, DC, MED, GAP and AGSP were used in the final version of HEVC-HR.

Table 6.5. BD-PSNR-YUV and BD-RATE results against HEVC-PVS using different intra-MI prediction modes

Prediction Mode	Order	BD-PSNR-YUV	BD-RATE
DC	1 to 4	0.25 dB	-10.72 %
MED	3	0.15 dB	-6.77%
GAP	7	0.28 dB	-12.12%
AGSP	9	0.29 dB	-12.87%

Table 6.6. BD-PSNR-YUV and BD-RATE results against HEVC-PVS using different LSP prediction mode configurations

LSP Order (M)	M_T	BD-PSNR-YUV	BD-RATE
3	*1 (Intra)*	*0.26 dB*	*-11.37 %*
	5 (Inter)	**0.31 dB**	**-13.10 %**
	9 (Inter)	0.31 dB	-13.29 %
5	*1 (Intra)*	*0.22 dB*	*-9.73 %*
	5 (Inter)	**0.32 dB**	**-13.42 %**
	9 (Inter)	0.43 dB	-13.98 %
7	*1 (Intra)*	*0.17 dB*	*-7.47 %*
	5 (Inter)	**0.30 dB**	**-12.55 %**
	9 (Inter)	0.32 dB	-13.50 %
9	*1 (Intra)*	*0.20 dB*	*-6.89 %*
	5 (Inter)	0.32 dB	-13.31 %
	9 (Inter)	0.34 dB	-13.69 %

Inter-MI prediction modes evaluation

Table 6.6 presents the experimental results for different configurations of the LSP-based modes described in Section 6.3. Two parameters were tested, corresponding to the LSP order (M) and the number of MIs used for training (M_T). By varying M, it is possible to compare the performance of an adaptive mode with prediction modes with similar orders, like MED, GAP and AGSP. The value M_T was tested for one, five and nine, which corresponds to use, respectively: the current MI for training (equivalent to an intra-MI prediction mode, as mentioned in Section 6.3); the current MI and the MI on the left, top, right and bottom of the current MI; and the current MI and the eight surrounding MIs.

From Table 6.6 it is possible to conclude that, regardless of the LSP order, when the training area (M_T) increases, the coding efficiency also increases. This is especially noticeable when using more than one MI for training. However, increasing the order does not always result into higher bitrate savings. This occurs because the use of higher LSP orders requires larger areas of reconstructed pixels, for LSP training. The size of the available training grows from the first frames to the last ones, affecting the quality of the training step. Thus, the use of higher LSP

orders will be more efficient only at later stages of the coding process, while using a lower order may be beneficial since an earlier stage of the coding process. Higher prediction orders and larger training areas also impact negatively (i.e., increase) on the computational complexity. In Table 6.6, the best three LSP based prediction methods in terms of bitrate savings vs. computational complexity are represented in bold (LSP3, LSP5 and LSP7, using five MIs for training. These modes were included in HEVC-HR. LSP9 modes were excluded, due to their high computational complexity.

HEVC-HR using Intra-MI and inter-MI prediction modes

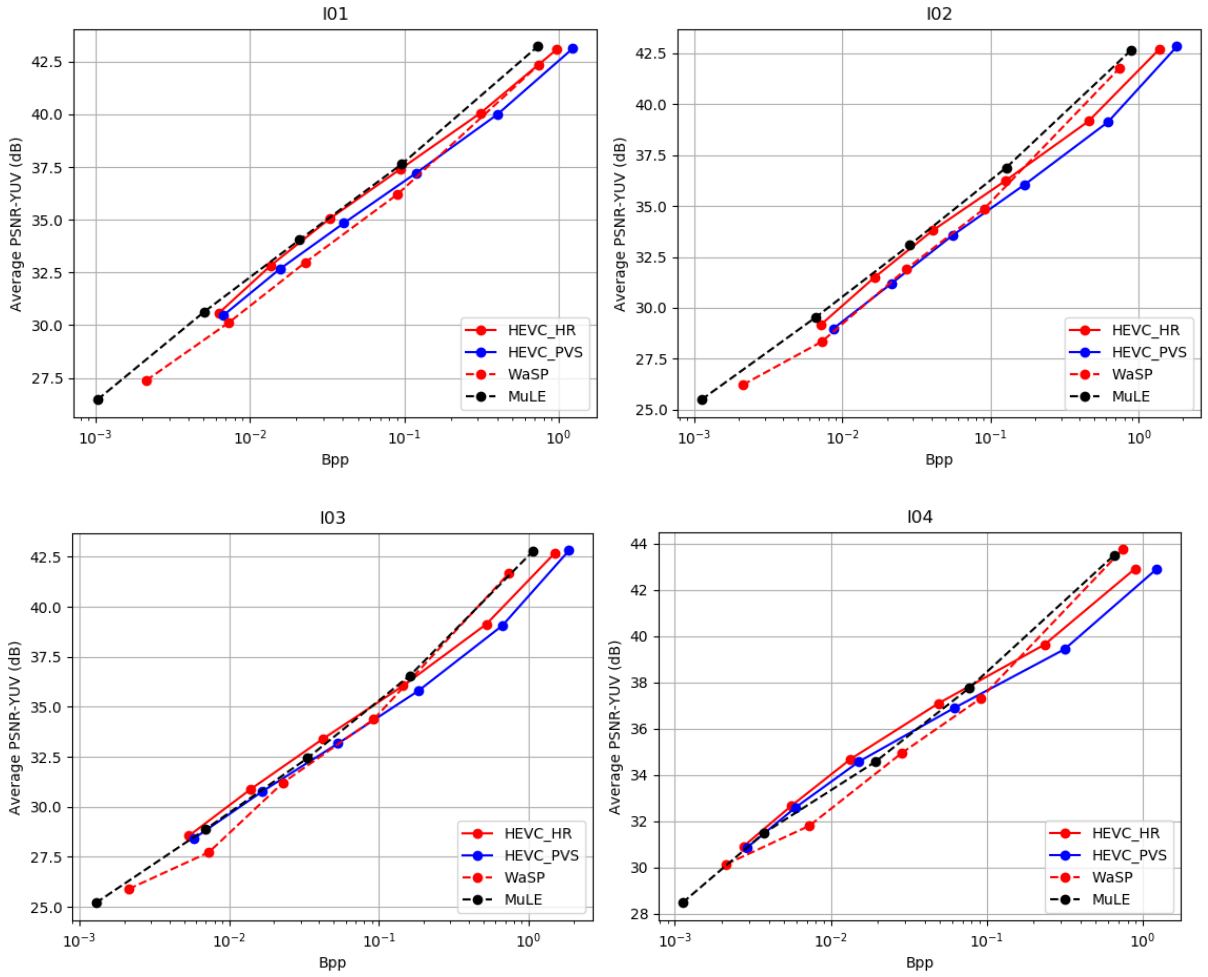
Table 6.7 presents the experimental results achieved by HEVC-HR when the prediction modes selected in the previous sections (DC, MED, GAP, AGSP, LSP3, LSP5 and LSP7) are jointly used; HEVC-HR using the intra-MI prediction modes only (DC, MED, GAP and AGSP); MuLE and WaSP LF image codecs in comparison with HEVC-PVS.

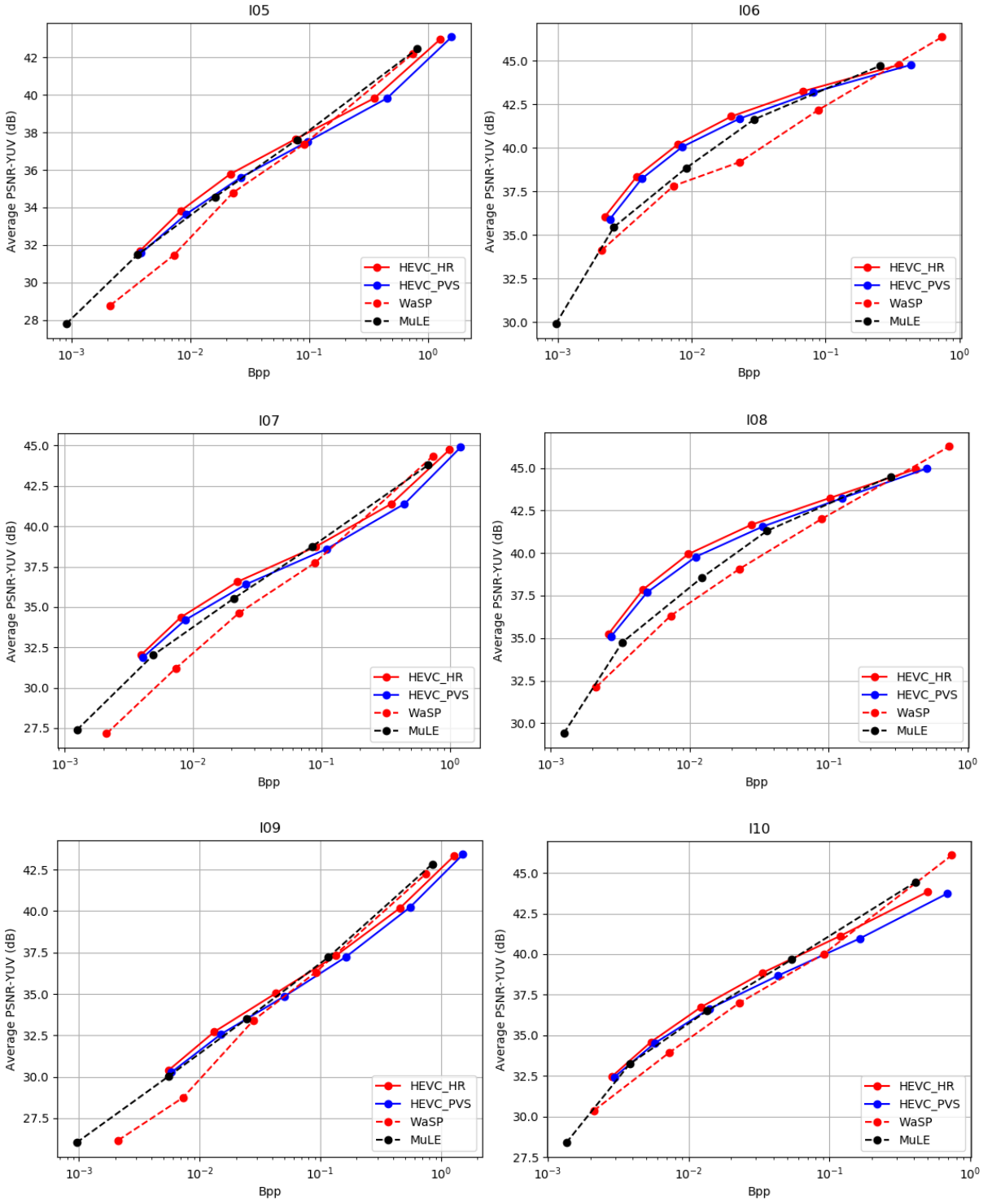
Table 6.7. BD-PSNR-YUV and BD-RATE results against HEVC-PVS using HEVC-HR, MULE and WASP codecs

Img.	**HEVC-HR** vs HEVC-PVS		**HEVC-HR (Intra-MI)** vs HEVC-PVS		MuLE vs HEVC-PVS		WaSP vs HEVC-PVS	
	BD-PSNR (dB)	BD-RATE (%)	BD-PSNR (dB)	BD-RATE (%)	BD-PSNR (dB)	BD-RATE (%)	BD-PSNR (dB)	BD-RATE (%)
I01	0.69	-24.65	0.48	-18.04	**0.97**	**-33.15**	-0.24	9.14
I02	0.92	-30.37	0.63	-21.72	**1.54**	**-45.55**	0.56	-23.07
I03	0.72	-25.34	0.50	-18.30	**0.81**	**-28.74**	0.25	-17.04
I04	**0.57**	**-25.49**	0.44	-20.20	0.48	-22.29	-0.06	-2.05
I05	**0.49**	**-23.40**	0.35	-17.48	0.40	-20.79	-0.19	1.93
I06	**0.33**	**-17.59**	0.25	-13.49	-0.58	31.16	-0.85	64.00
I07	**0.50**	**-20.44**	0.36	-15.30	0.23	-12.20	-0.79	25.53
I08	**0.36**	**-17.68**	0.27	-13.58	-0.70	38.08	-1.05	39.75
I09	0.49	-19.00	0.31	-12.46	**0.49**	**-19.94**	-0.01	-7.13
I10	**0.60**	**-24.91**	0.48	-20.35	0.48	-21.34	-0.07	9.44
I11	**0.37**	**-18.20**	0.26	-12.92	0.07	-5.32	-0.99	36.33
I12	**0.69**	**-25.23**	0.45	-17.26	-0.19	5.02	-1.02	34.86
Avg..	**0.56**	**-22.69**	0.40	-16.76	0.33	-11.26	-0.37	14.31

From Table 6.7 it is possible to observe that HEVC-HR consistently outperforms HEVC-PVS in terms of coding efficiency. An average of 22.69% of bitrate savings is achieved by using multiple prediction modes, which is considerably higher than the best performance of an individual prediction mode, i.e., LSP5, with 13.98%. Additionally, HEVC-HR using only intra-

MI prediction modes, achieved 16.76% bitrate savings over HEVC-PVS. From this table it is also possible to observe that MuLE exhibits an average bitrate savings, of 11.26%, while WaSP exhibits an average increase in bitrate of 14.31%, relatively to HEVC-PVS. This means that the proposed HEVC-HR is on average more efficient than both MuLE and WaSP for the twelve LF images. Moreover, from the twelve test LF images used, HEVC-HR is more efficient than MuLE for eight out of the twelve images. MuLE is more efficient than HEVC-HR for images I01, I02, I03 and I09. The RD curves for the four LF image codecs when encoding the twelve EPFL LF dataset images are shown in Figure 6.8. From these RD curves it is possible to observe that HEVC-HR is able to outperform HEVC-PVS consistently for every image. Additionally, HEVC-HR tends to be more efficient than MuLE for lower bitrates.




I05
I06
I07
I08
I09
I10
Average PSNR-YUV (dB)
Bpp
HEVC_HR
HEVC_PVS
WaSP
MuLE

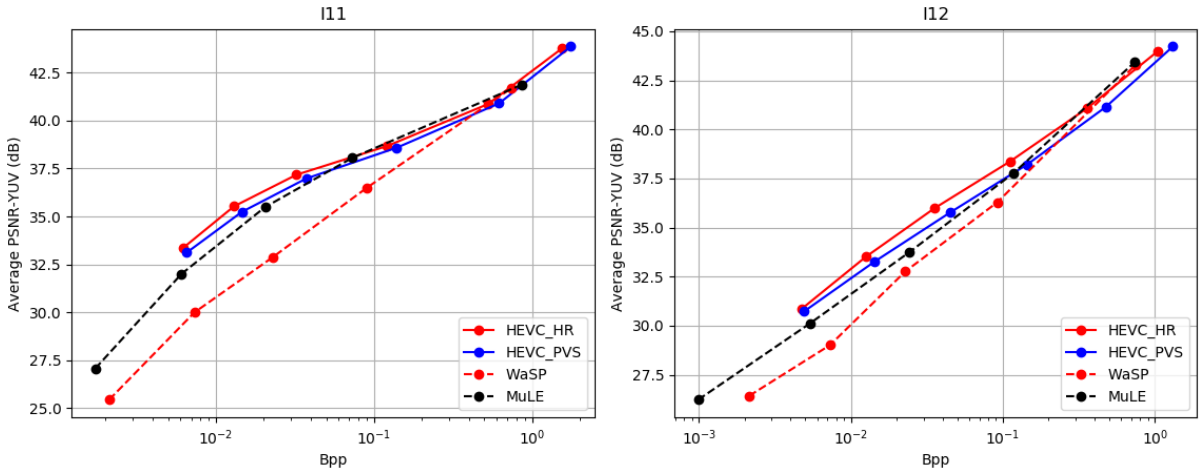


Figure 6.8. RD Curves comparing the proposed hybrid representation LF coding approach (HEVC-HR) and HEVC-PVS, MuLE and WaSP.

In order to further analyze the usefulness of each prediction mode in HEVC-HR, the average prediction mode usage across the six QPs is shown in Table 6.8. The values in bold correspond to the most used prediction modes, while the values in italic correspond to the least used prediction modes. It is possible to observe that most of the pixels are encoded using inter-SAI prediction modes, because the inter-view redundancy is very high in this type of LF content. However, when analyzing the prediction mode usage for intra-MI and inter-MI prediction, which include the new seven modes, it is possible to conclude that, for most images, the new modes are more often used than the intra-SAI modes, i.e., DC, Planar and the 26 remaining directional modes. These statistics allow to conclude that exploiting the intra- and inter-MI redundancy results in more coding efficiency than exploiting the spatial redundancy within each SAI.

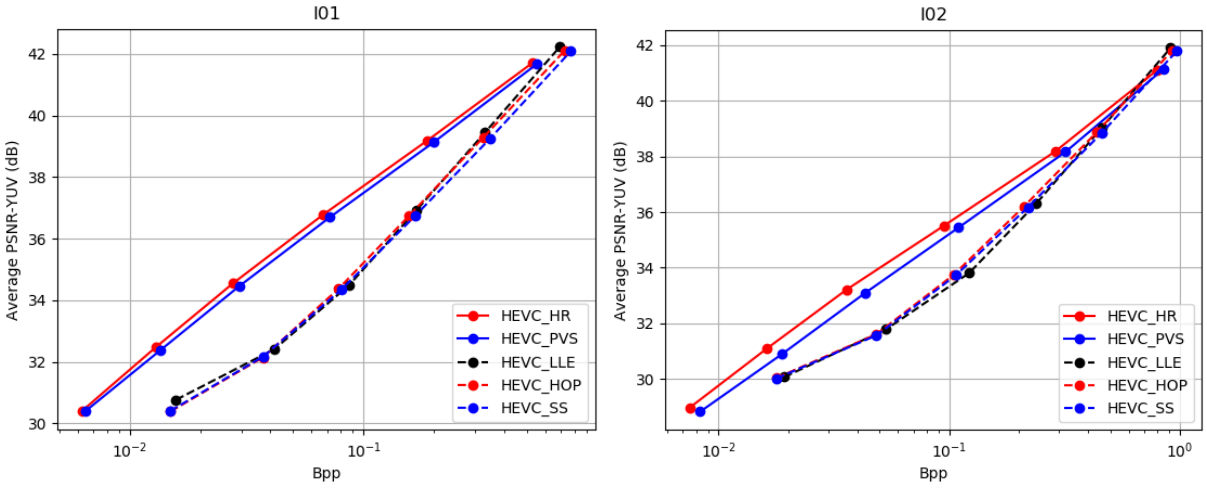
Amongst the new prediction modes, proposed in this Thesis, the most used prediction mode is LSP7, which verifies the assumption made about the usefulness of LSP-based prediction modes. LSP7 when tested individually is not as efficient as LSP5, because of the higher requirements in terms of training area in the initial phase of encoding process. Nevertheless, the use of three LSP-based prediction modes, with different orders and, therefore, different requirements in terms of training area, allows the encoder to choose the more suitable prediction mode for every phase of the coding process.

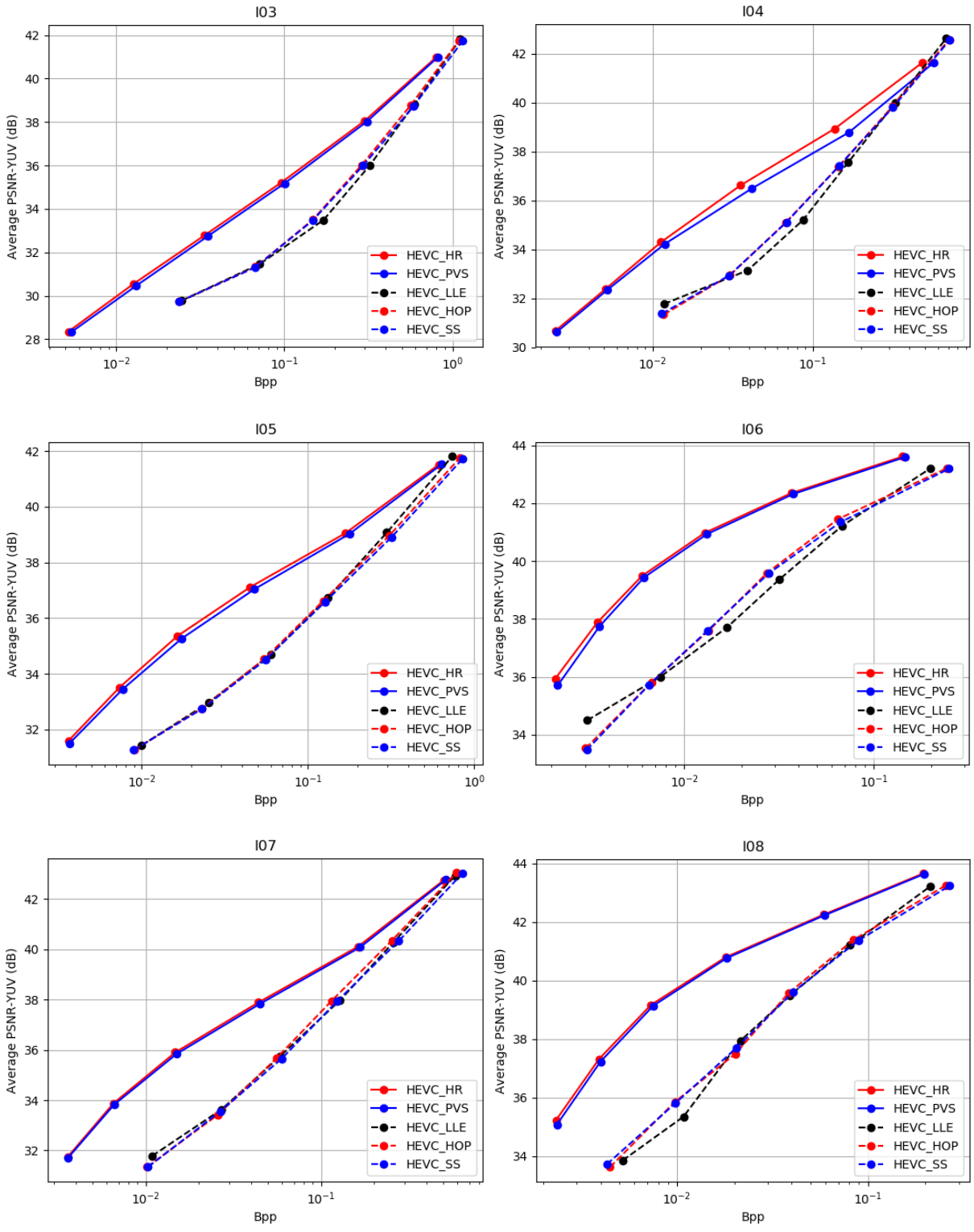
Table 6.8. Average prediction mode usage across the six QPs, in percentage of pixels for the HEVC-HR codec

Img.	Inter-SAI	Intra-SAI	Intra-MI				Inter-MI		
			DC	MED	GAP	AGSP	LSP3	LSP5	LSP7
I01	**72.3**	1.5	3.7	2.6	*1.4*	2.4	3.2	5.3	**7.5**
I02	**73.0**	1.4	3.1	2.2	*1.2*	2.2	3.3	4.9	**8.8**
I03	**78.7**	1.3	2.3	1.5	*0.8*	1.9	2.5	4.0	**7.1**
I04	**75.5**	*1.3*	3.3	1.8	1.3	2.1	2.4	4.5	**7.8**
I05	**76.0**	2.0	3.5	2.4	*1.0*	1.8	3.2	4.3	**5.8**
I06	**70.6**	**6.9**	5.5	3.0	2.2	*1.6*	3.0	3.4	3.7
I07	**79.1**	2.2	2.7	2.0	*1.4*	1.5	2.5	3.6	**5.0**
I08	**68.9**	**7.9**	5.0	3.3	2.0	*1.8*	3.1	4.0	4.1
I09	**69.8**	2.3	3.9	3.7	*1.1*	1.8	3.2	6.3	**8.0**
I10	**75.6**	*1.4*	4.1	2.4	1.8	2.2	2.9	3.9	**5.7**
I11	**69.2**	4.0	4.4	**4.9**	2.5	*1.6*	4.1	4.8	4.6
I12	**70.0**	2.2	3.3	3.3	*1.5*	2.2	3.3	6.2	**7.8**

Experimental evaluation for YUV 4:2:0 8 bit color format

In order to compare HEVC-HR with MI data representation LF coding approaches, a set of similar tests using the YUV 4:2:0 8 bit color format were performed, since the available implementations of HEVC-SS, HEVC-HOP and HEVC-LLE are only compatible with the YUV 4:2:0 8 bit color format. This allows for a fair comparison between PVS, MI and the proposed hybrid approach for LF image coding. The RD curves for the twelve EPFL LF dataset images are shown in Figure 6.9. These RD curves are used to compare all the codecs listed in Table 6.3 using the YUV 4:2:0 8 bit color format.




I03
I04
I05
I06
I07
I08
Average PSNR-YUV (dB)
Bpp
HEVC_HR
HEVC_PVS
HEVC_LLE
HEVC_HOP
HEVC_SS

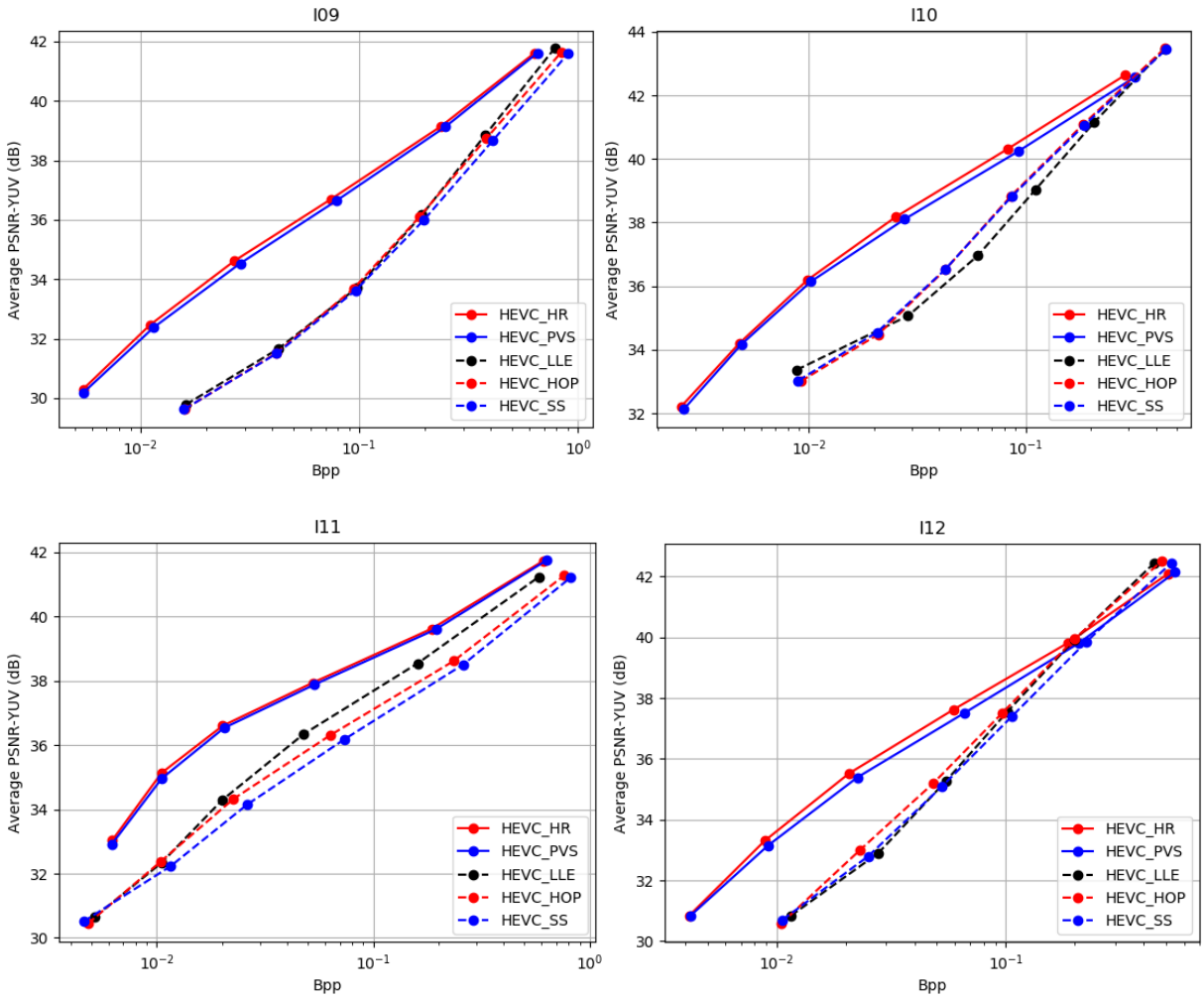


Figure 6.9. RD performance of the proposed hybrid representation LF coding approach (HEVC-HR) against the 4DLF-MI and 4DLF-PVS representation approaches for selected LF test images.

The RD curves in Figure 6.9 show that coding approaches based on 4DLF-PVS outperform approaches based on 4DLF-MI. This is explained by the fact that the inter-view redundancy between SAIs is very high and easily exploited by the inter-prediction tools of HEVC.

Although HEVC-PVS is more efficient than approaches based on 4DLF-MI, the proposed HEVC-HR, based on a hybrid LF data representation, is able to achieve the highest coding efficiency, outperforming HEVC-PVS. The achieved average bitrate savings when compared to HEVC-PVS, for the twelve LF images in the YUV 4:2:0 8 bit color format, is 9.36%. This shows that the proposed hybrid data representation and prediction modes are able to increase the coding efficiency, regardless of the color format.

Computational complexity assessment

The computational complexity of the tested codecs is presented in Table 6.9. The runtime values shown in this table were measured while encoding and decoding the image I01, with a $QP = 22$, for all the listed HEVC-based codecs. MuLE was tested using a λ value of 270 and WaSP was tested using a target bitrate of 0.75 bpp. These tests were performed using a PC equipped with an Intel Core i7 CPU 4790K@4.0GHz and 32GB of RAM, running Ubuntu 16.04.

Table 6.9. Codec single thread computational complexity comparison

Codec	Encoder		Decoder	
	Run Time (hours)	vs HEVC-PVS (ratio)	Run Time (seconds)	vs HEVC-PVS (ratio)
YUV 4:2:0 8bpp				
HEVC-PVS	0.34	-	1.22	-
HEVC-SS	4.43	13.18	372.10	304.01
HEVC-HOP	6.50	19.32	396.81	324.19
HEVC-LLE	11.74	34.90	1011.89	826.71
HEVC-HR	22.18	65.25	326.10	267.30
HEVC-HR (Intra-MI)	**0.38**	**1.12**	**5.25**	**4.30**
YUV 4:4:4 10bpp				
HEVC-PVS	0.54	-	3.21	-
MuLE	0.15	0.28	18.19	5.67
WaSP*	**0.14**	**0.26**	38.46	11.98
HEVC-HR	24.14	44.70	3293.75	1026.09
HEVC-HR (Intra-MI)	0.59	1.09	**11.53**	**3.58**

*using multithread (8 threads)

Although the coding efficiency of the proposed HEVC-HR is higher than all the tested benchmarks, this comes at the expense of a higher computational complexity. From Table 6.9 it is possible to see that HEVC-HR takes a much longer time to encode and decode the same LF image. Note however that, none of the implementations, including the HEVC-HR and the 4DLF-MI coding approaches, are optimized. It is also possible to see that when testing HEVC-HR using only the intra-MI prediction modes, the computational complexity is only marginally higher than HEVC-PVS, while still being on average more efficient than MuLE and WaSP. From this result it is possible to conclude that the computational complexity increase of HEVC-HR relative to HEVC-PVS comes mostly from the inter-MI prediction modes, therefore these prediction modes would benefit from a more optimized implementation or parallelization.

6.5. Final remarks

In this chapter, a new hybrid LF data representation paradigm for LF data coding is presented. A HEVC-based codec implementation is described, as well as a set of pixel-based prediction modes to efficiently compress LF images. The hybrid LF data representation comprises both MI- and SAI-based representations to enhance the reference domain for the prediction modes. To efficiently exploit the intra-MI redundancy of each MI, some pixel-based prediction methods, such as DC, MED, GAP and AGSP were adapted to the proposed codec. Additionally, in order to exploit the inter-MI redundancy, efficient pixel prediction modes based on LSP using different order values were proposed.

The proposed HEVC-HR codec was evaluated against state-of-the-art codecs. When compared with HEVC-PVS, for the YUV 4:4:4 10 bit color format, an average bitrate saving of 22.69 % was achieved, while for the YUV 4:2:0 8 bit color format, the average bitrate saving was 9.36%. Additionally, the RD curves show that the proposed HEVC-HR was also able to outperform MI-based benchmarks such as HEVC-SS, HEVC-HOP and HEVC-LLE using the YUV 4:2:0 8 bit color format, for all used test images.

Approaches such as MuLE and WaSP, which (at the time of writing) are being considered for the JPEG Pleno standard, were also used as benchmarks. Such approaches were outperformed by the proposed HEVC-HR solution, which only achieve overall bitrate savings over HEVC-PVS of 11.26% and -14.31%, respectively.

Chapter 7. Light field image coding with viewpoint scalability and random access

This chapter proposes a LF image codec based on PVS that presents state-of-the-art coding efficiency, whose main contribution is to provide viewpoint scalability and random access functionalities. In this context, a new scanning order is proposed, combined with an optimized reference picture selection to allow viewpoint scalability over six layers. The viewpoint random access functionality can be used according to the application requirements. However, it implies a tradeoff between viewpoint random access capabilities and coding efficiency. This is achieved by adjusting the viewpoint dependencies through the use of several proposed coding profiles.

The remainder of this chapter is organized as follows: Section 7.1 reviews the available solutions for LF viewpoint scalability and random access; Section 7.2 presents the proposed PVS-based LF image codec; in Section 7.3 viewpoint scalability is added to the proposed codec; Section 7.4 presents the proposed control parameters that allow viewpoint random access; Section 7.5 presents the experimental results and, finally, Section 7.6 concludes the chapter with some final remarks.

7.1. Related work on viewpoint scalability and random access

This section presents several schemes to encode LF images that allow some viewpoint scalability and random access.

7.1.1. Viewpoint scalability

In [115], the authors started exploring scalability functionalities for LF, by proposing a two-layer LF coding approach for the focused LF camera model. It uses a LF representation that

consists of a sparse set of MIs and associated disparity maps. Based on the sparse set of MIs and the associated disparity maps (first layer), a reference prediction LF image is obtained through a reconstruction method that relies on disparity-based interpolation and inpainting. This reconstructed LF image is then used to encode the original LF image (second layer), by encoding the prediction residue. This approach was later extended [116] with a third layer of scalability and the use of lossy encoded disparity maps to improve the coding efficiency, in contrast with the lossless transmission of the disparity maps used in the previous approach.

In order to increase backward compatibility with legacy displays, the authors in [117] propose a three-layer approach. A certain number of viewpoints was assigned to each layer, i.e., the first layer encodes the central view, the second layer encodes stereo or multiview and the third layer encodes the full LF image. In order to increase the coding efficiency inter-layer prediction was used to exploit the redundancy between layers. This work was recently extended [118] to a higher number of scalability layers, allowing to improve the coding efficiency by using an exemplar-based algorithm for texture synthesis.

Also, the techniques based on structural key views (see Section 3.4.3), may be considered as scalable approaches. In these cases, several authors propose to encode only few viewpoints and use additional information to generate the remaining ones. These types of approaches allow for viewpoint scalability, because the LF image is in fact encoded using two scalable layers. The base layer comprises the SKVs and the enhancement layer includes the non-SKVs.

7.1.2. Viewpoint random access

In order to provide viewpoint random access, the LF coding algorithm needs to select the dependencies between viewpoints. The more constrained these dependencies are, the higher the viewpoint random access capabilities, however, the coding efficiency tends to decrease. In [119], the authors propose to eliminate prediction at the encoder, therefore eliminating viewpoint dependency, by using Wyner-Ziv coding for compressing LF images. This work was extended in [120] by using SP-frame predictive encoding. More recently, in [121], the authors decompose 15×15 viewpoints into 25 groups of viewpoints and allocate four different dependency levels to each group. Finally, in [122], it was proposed a MV-HEVC based coding solution, that allows diagonal viewpoint prediction instead of exclusively allowing horizontal and vertical viewpoint prediction. Experimental results show that allowing diagonal viewpoint

prediction provides a good compromise between coding efficiency and viewpoint random access when compared to algorithms that are exclusively based on horizontal and vertical viewpoint prediction.

7.2. Optimized reference picture selection

As discussed in Section 3.4.1, the PVS format is comprised of a sequence of SAIs, whose inter-SAI redundancy may be efficiently exploited by the HEVC inter-prediction tools. The existence of redundancy between the selected viewpoints is crucial for the performance of the prediction techniques. Since the SAIs scanning order plays an important role on the prediction performance, a reference picture selection (RPS) method was developed to optimize it. This method is able to implicitly signal to the decoder the scanning order used prior to be encoded. Once the encoder and the decoder are aware of the scanning order, the RPS can be optimized.

7.2.1. Generic scanning order

This PVS-based LF image coding approach is generic in terms of scanning order, because this information can be signaled to the decoder. Therefore, the spiral scan was adopted as it was seen in Section 3.4.1 is more efficient than the raster and serpentine scan. Figure 7.1 shows the spiral scan being applied to an $N \times N$ matrix of viewpoints, where $j = [0, 1, \dots, N-1]$ and $i = [0, 1, \dots, N-1]$ are the vertical and horizontal axis spatial positions, respectively, for each viewpoint in the matrix of viewpoints. In this case the decoder can determine the spatial position of each viewpoint based on the viewpoint order in the spiral scanning.

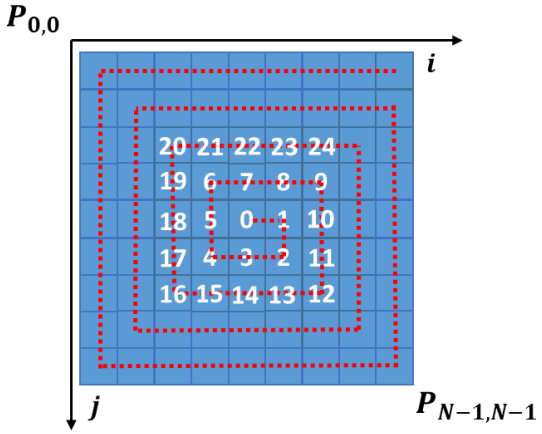


Figure 7.1. Spiral PVS scanning order applied to a N×N matrix of viewpoints, where P represents the position of each viewpoint.

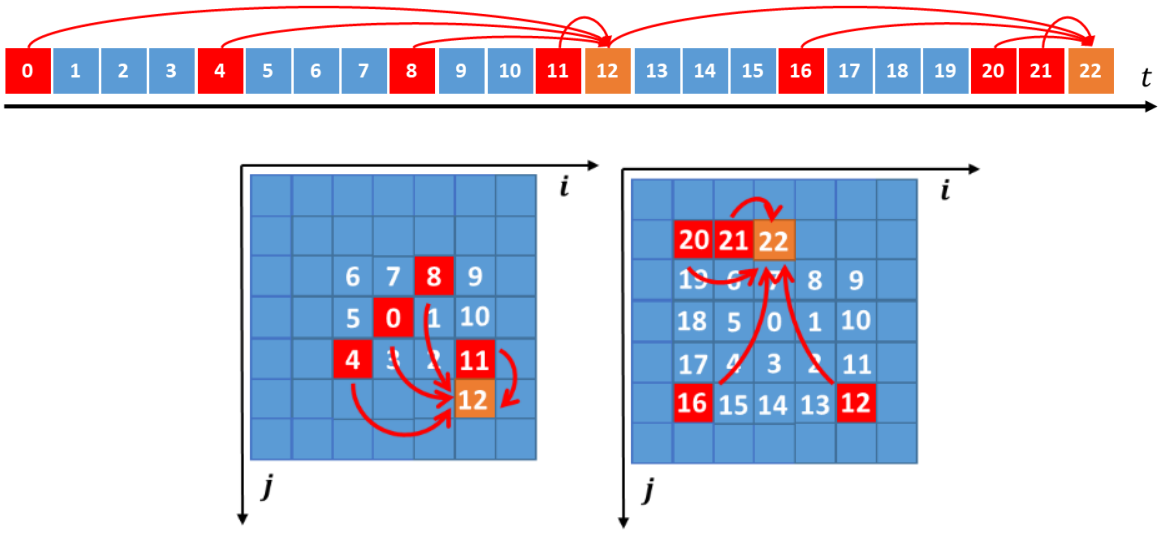


Figure 7.2. RPS for frames 12 and 22 when using the HEVC Low Delay configuration, represented in the temporal domain (top) and corresponding N×N matrix of viewpoints (bottom).

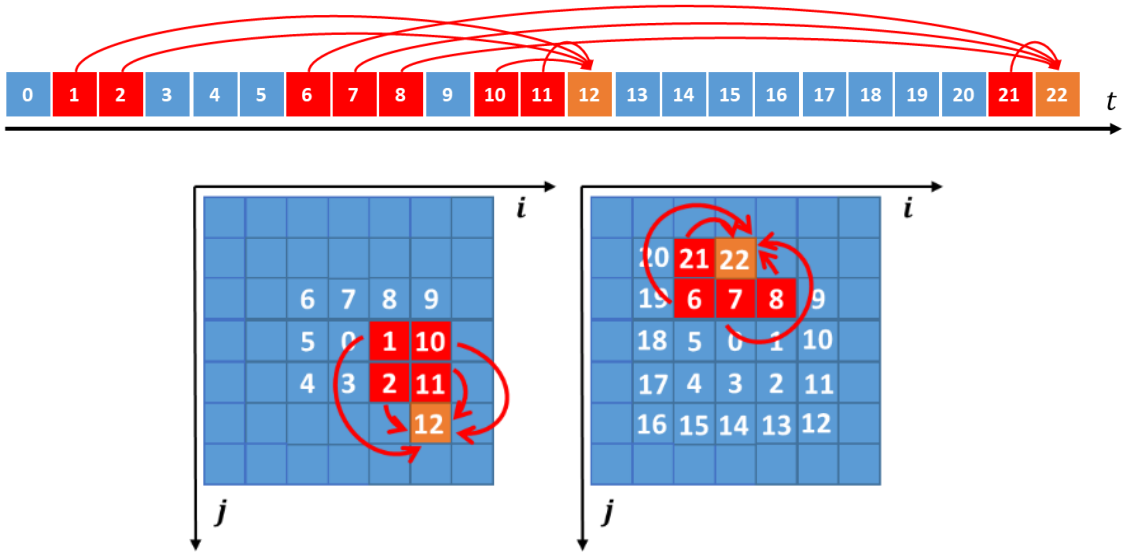


Figure 7.3. Optimized RPS for frames 12 and 22 represented in the temporal domain (top) and the corresponding N×N matrix of viewpoints (bottom).

7.2.2. Optimized RPS

In the common application of a PVS-based LF coding a default configuration such as the HEVC Low Delay configuration, is used [64]. Consequently, the inherent 2D spatial location of each viewpoint is not considered and the used reference pictures are not selected based on their

expected correlation with the current viewpoint. This situation can be observed in Figure 7.2 for frames 12 and 22, represented in the temporal domain and organized in the $N \times N$ matrix of viewpoints. Indeed, there is a regular pattern, which is defined by HEVC Low Delay configuration, being applied in the temporal domain in terms of RPS. However, the RPS is not optimal for most of the reference pictures of the $N \times N$ matrix of viewpoints. In the example of Figure 7.2, when encoding frame 22, frames 6, 7 and 8 are closer, in terms of spatial position, when compared to frames 20, 16 and 12.

To create an optimized RPS, the Euclidean distance, d, between the spatial positions of R reference pictures, P_{ji}^{r}, and the current viewpoint, P_{ji}, should be minimized. The Euclidean distance between the current viewpoint and the remaining available encoded viewpoints is calculated by:

$$d(P_{ji}, P_{ji}^{r}) = \sqrt{(j - j_r)^2 + (i - i_r)^2}, \tag{7.1}$$

where $r = [0,1, \dots, R-1]$. Once the R closer reference pictures (in terms of Euclidean distance) to the current viewpoint are found, the selected reference pictures are organized in an ascending order of distance, in the RPL of each viewpoint that is being encoded. When two or more reference pictures have the same Euclidean distance, the one with the lowest frame number is selected first. Figure 7.3 shows the optimized RPS for $R = 4$, after minimizing the Euclidean distance as in (7.1), for the examples of frame 12 and 22, previously shown in Figure 7.2. From Figure 7.3 it is possible to see that, although in the temporal domain there is no longer a regular pattern, in the $N \times N$ matrix of viewpoints the RPS minimizes (7.1), potentially maximizing the correlation between viewpoints and, consequently, the coding efficiency.

7.3. Viewpoint scalability

In this section, the coding approach described in Section 7.2 is modified to accommodate viewpoint scalability, which allows the codec to enable LF content to be captured and displayed using various types of devices. These devices range from standard 2D cameras and displays, up to full-fledged LF cameras and displays. The following sections present the features that are enabled by viewpoint scalability, as well as the proposed scalable scanning order and the scalability layers.

7.3.1. LF viewpoint scalability features

Viewpoint scalability allows the LF content to be represented in several layers, where each layer comprises a group of viewpoints, ultimately allowing compatibility with capture and display devices with different capabilities, like spatial resolution, angular resolution and processing power. Also, network conditions may also vary, regardless of the capabilities of the device. Therefore, viewpoint scalability can be advantageous for several steps of the transmission pipeline like the capturing and encoding steps:

- **Support for legacy capturing devices**: The first two layers should include the central view (base layer) and a small number of horizontal and vertical views (second layer). This allows the scalable representation to be compatible with 2D and 3D/Stereo capturing devices;
- **Support for LF captured from both lenslet LF cameras and HDCAs**: When both lenslet and HDCA LFs are represented by viewpoints the only significant difference is the baseline between the several viewpoints.
- **Support for scalable coding profiles**: The number of scalability layers to be encoded and transmitted can be selected based on several criteria, such as: the available processing power, the available storage space and the network conditions.

Additionally, viewpoint scalability can also be advantageous for decoding and displaying steps:

- **Support for legacy display devices**: Compatibility with legacy, non-LF displays, e.g., 2D and 3D/Stereo displays, where the first layers include the central view and the first side views.
- **Support for scalable decoding**: Each consecutive layer is decoded cumulatively, therefore after each decoded layer the content can be displayed. The angular resolution fidelity is increased as the remaining layers are decoded.
- **Support for LF displays with varying capabilities**: Each layer will require additional capabilities in terms of angular resolution and processing power. Therefore, the number of displayed layers can be adaptively adjusted based on both criteria;

- **Support for the newest LF displays**: LF displays that allow to show the full LF content are able to request all the layers to be received and decoded.

7.3.2. Proposed scalability layers and scalable scanning order

Various combinations of scalability layers can be used to implement the mentioned features. Regardless, it was adopted the hierarchical structure in [42], which provides the required distribution of viewpoints throughout the coding process for the viewpoint scalability features mentioned in this section. The proposed scalability structure is shown in Figure 7.4, where the orange and the yellow blocks represent the viewpoints from the current and previous layers, respectively, and the blue blocks stand for the viewpoints not considered yet.

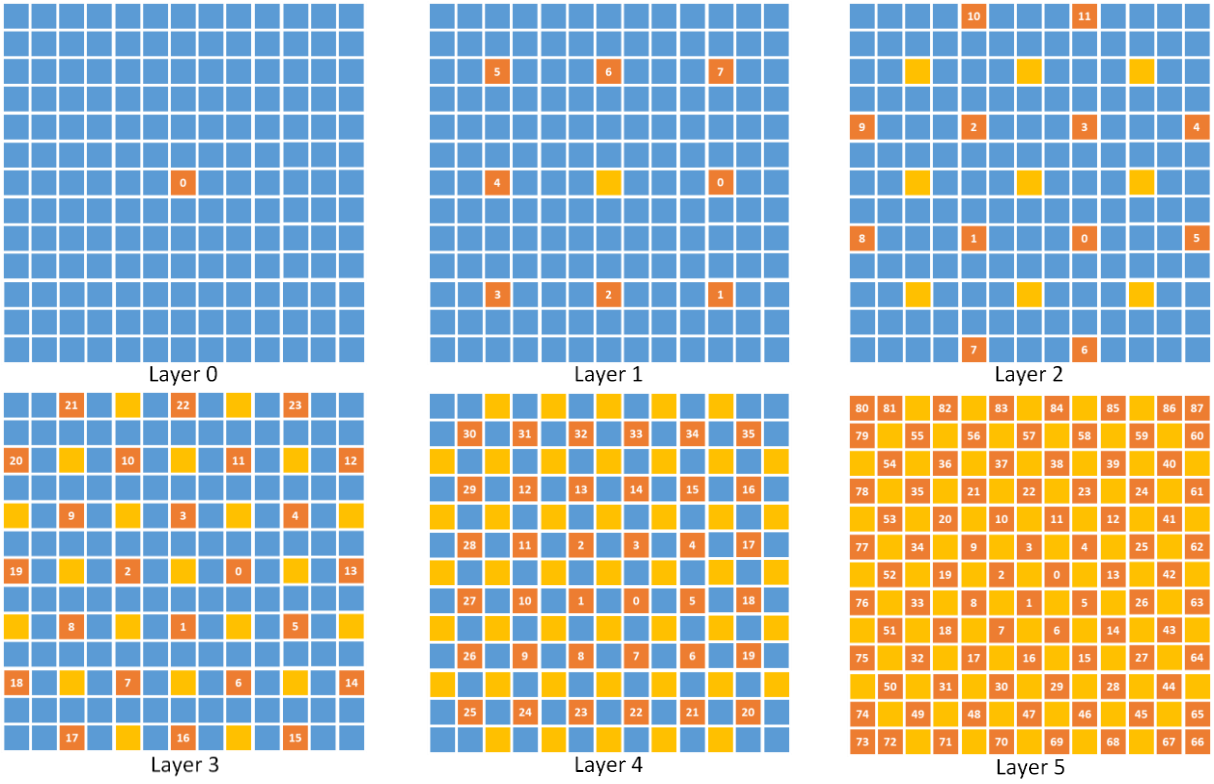


Figure 7.4. Proposed LF scalability layers with respective coding order per layer.

The viewpoint distribution across the several layers, roughly follows a $(2n+1) \times (2n+1)$ pattern, where n is the layer number. Layer 0 allows for the central view to be encoded/decoded independently for compatibility with 2D displays. Layer 1 allows a 3×3 LF to be displayed, or a pair of views can be selected to be displayed in a stereo display. After decoding Layers 2, 3 and 4, the resulting LF is roughly a 5×5, 7×7 and 9×9 LF, respectively. These intermediate

layers allow to have some flexibility in terms of angular resolution and processing capabilities, as well as when the network condition changes. Finally, Layer 5 includes the remaining viewpoints that represent the full 13×13 LF image.

In order to encode and decode several scalability layers, a scalable scanning order is proposed. Since the optimized RPS minimizes the Euclidean distance to the viewpoint that is being encoded, this means that it will adapt to the used scanning order.

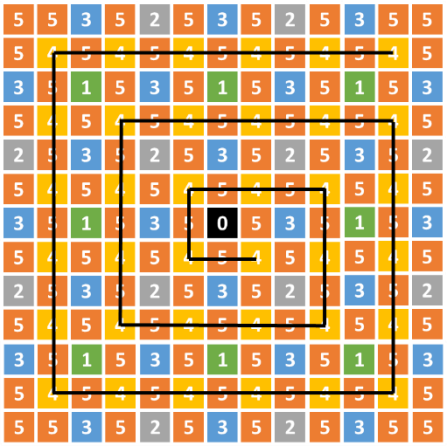


Figure 7.5. Scalability layer mask.

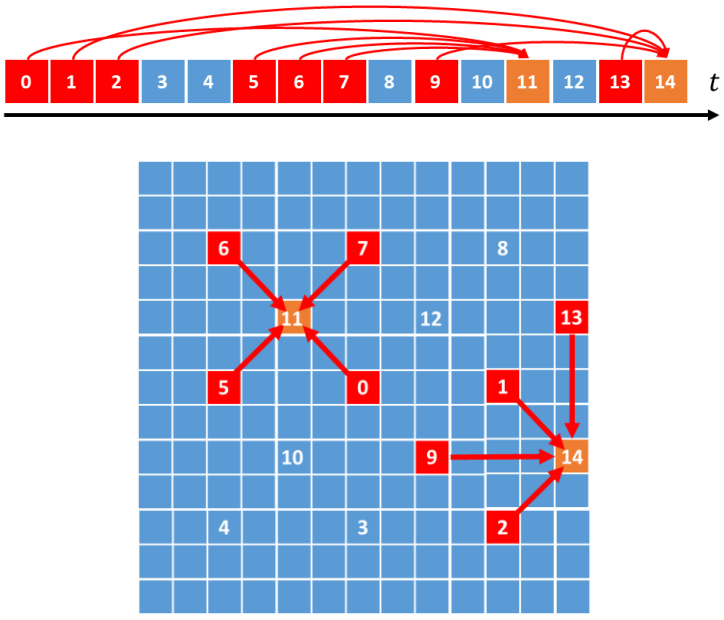


Figure 7.6. Optimized RPS when applied to the proposed LF scalable spiral scanning order.

The proposed LF scalable spiral scan is shown for each individual layer in Figure 7.4. As it is shown, the number in each viewpoint corresponds to the scalable spiral scan being applied to each scalability layer.

A spiral scanning order is applied to each individual scalability layer, as shown in Figure 7.5. Each number inside an element of the mask represents the layer that each viewpoint is assigned to. The scalable spiral scan is generated by applying the spiral scan to viewpoints within each layer. For example, the spiral marked in black in Figure 7.5 corresponds to the Layer 4 spiral scan.

When encoding the LF image using the proposed scalable spiral scanning order, the optimized RPS coding technique will adapt to the new scanning order. It will provide the most adequate reference frames for each situation, as can be seen in Figure 7.6 for frames 11 and 14, by minimizing the Euclidian distance (7.1).

7.4. Viewpoint random access

The viewpoint random access functionality allows the codec to generate a bitstream that facilitates viewpoint navigation and user interaction. The following sections present how to accommodate viewpoint random access with the proposed viewpoint scalability, as well as the proposed random access profiles.

7.4.1. Viewpoint random access features

In video coding algorithms, random access points are used to facilitate the interaction with the video sequence. This way it is possible to navigate the video sequence without having to decode the entire bitstream. In such case, I-frames are used as random access points because they can be decoded independently from the remaining frames, as only intra prediction modes are used.

In the case of a LF, the navigation can be performed using a 2D/3D display, a LF display or even a HMD [123]. Therefore, there is a vast number of possibilities because during visualizing the user shall be able to select the desired viewpoint. But, due to inter-viewpoint predictions used in the coding process several dependencies are created, which increase the number of decoded viewpoints in order to visualize a specific viewpoint. The aim of the proposed approach

in this chapter is to minimize the number of viewpoint dependencies, i.e., to maximize viewpoint random access, allowing to:

- **Improve LF navigation efficiency**: the number of necessary viewpoints to decode the desired viewpoints is reduced;
- **Reduce decoding delay**: since less viewpoints are necessary; the decoding time may be reduced;
- **Reduce computational complexity**: the computational complexity is also reduced, facilitating cases where the available processing power is limited.

Although maximizing viewpoint random access presents several advantages, this will interfere with the coding efficiency. Thus, depending on the application, it is possible to have a tradeoff between viewpoint random access and coding efficiency.

7.4.2. Proposed random access profiles

The measurement of the random access capability will be accessed using the random access penalty (RAP) metric [35]:

$$RAP = \frac{\#\ encoded\ bits\ required\ to\ access\ a\ RoI}{\#\ encoded\ bits\ to\ decode\ the\ full\ LF}, \tag{7.2}$$

The region of interest (RoI) can have different representations, depending on the application and coding algorithm, e.g., specific viewpoints or specific pixels. In this case, since the coding algorithm uses a viewpoint-based representation, the RoI corresponds to a specific viewpoint. The RAP for a specific viewpoint is the ratio between the number of bits required to decode that viewpoint, including the number of bits required to decode the reference viewpoints used to encode it, and the total number of bits to encode the full LF, as defined by (7.2). Since these reference viewpoints may also have other dependencies, it may happen that the full LF needs to be decoded in order to decode the RoI, which results in a $RAP = 1$. However, if only part of the LF image needs to be decoded, then $0 \leq RAP < 1$. Since the RAP depends on the selected RoI, in order to increase the fairness of the metric, only the maximum value will be considered, i.e., the viewpoint that requires the largest number of bits to be decoded.

The main factor that influences the RAP is the amount of inter-viewpoint dependencies created during the coding process. In order to manage these inter-viewpoint dependencies three control parameters are proposed:

- **RPL size**: Since by increasing the RPL size the maximum number of reference pictures increases, it is likely that the coding efficiency improves, until a certain limit. Consequently, since more inter-viewpoint dependences are created, the RAP increases as well;
- **Maximum scalability layer**: This parameter limits the depth of the scalability layer where viewpoints can be used as reference pictures. As shown in Figure 7.7, when this value is two, only the first nine viewpoints are used as reference pictures for the remaining viewpoints, which correspond to Layers 0 and 1. A lower coding efficiency and RAP are expected by using just part of the scalability layers as possible references. In Figure 7.7, several examples are shown, where the maximum scalability layer is adjusted as: six (all), four, three and two.
- **Viewpoint region**: When the user is navigating the LF image, either through a 2D/3D display (LF display with different capabilities) or an HMD, it is likely that certain spatial regions of the LF will be more visualized than others [124]. This way, several profiles are suggested in Figure 7.8, relying on different configurations of spatial regions and number of intra frames within the same LF. The red lines represent the limit of the reference picture's region. The yellow blocks show the viewpoints that belong to more than one region, e.g., in the "4-Regions 1-Intra" scenario, the block to the left of the I frame (central viewpoint) belongs to Region C and B. Such restrictions limit the choice of the reference pictures of each viewpoint to a spatial region. But, it ensures, for example in the case of the "2-Regions 1-Intra" scenario, that to visualize the upper viewpoints the lower viewpoints do not need to be decoded. When more than one intra frame is used, e.g., in the "5-Regions 5-Intra" scenario, there is an exclusive partition of the LF, which means that region A can be decoded independently from the remaining regions.

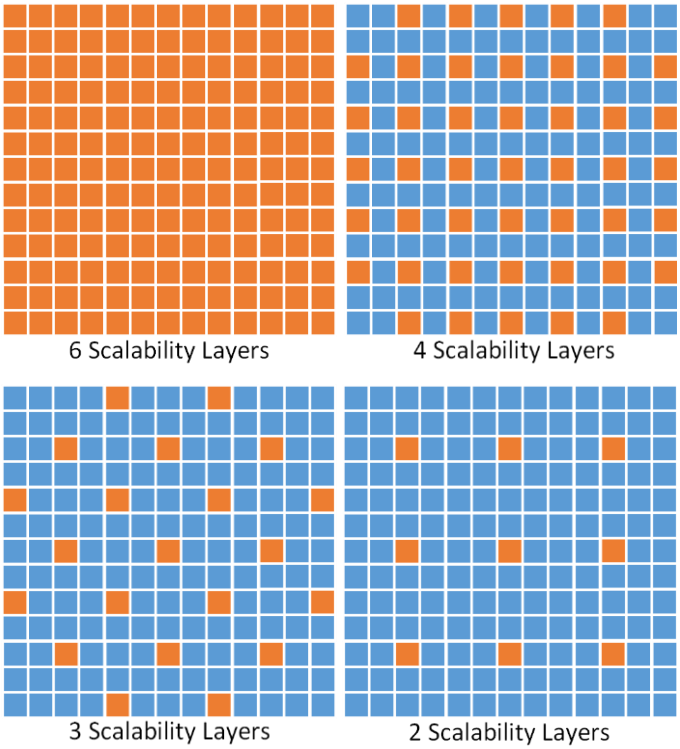


Figure 7.7. LF scalability layer dependency scenarios: the orange blocks represent the viewpoints belonging to the scalability layers that can be used as reference pictures.

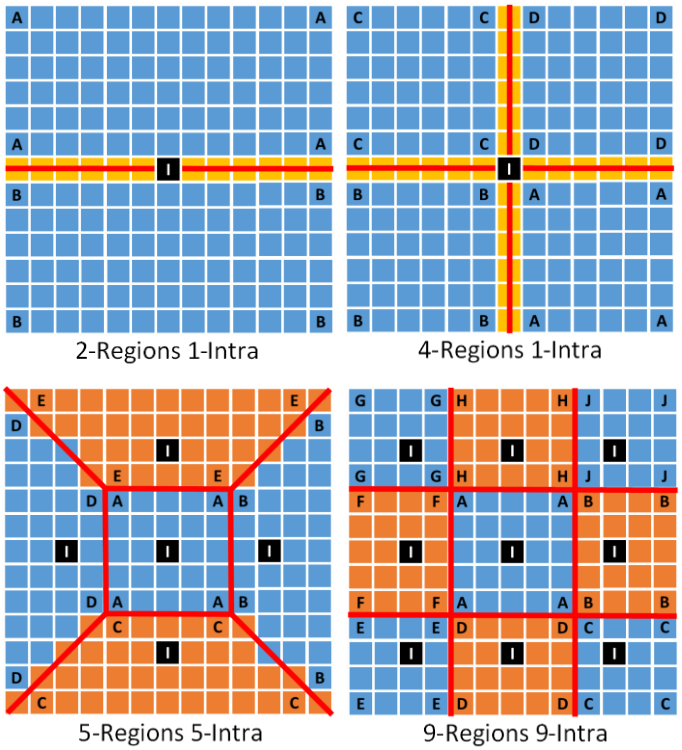


Figure 7.8. LF scalability viewpoint region scenarios: the suggested region separation using different number of regions and I frames per LF image.

7.5. Experimental results

In this section the performance of the proposed LF coding solution is evaluated against several state-of-the-art LF coding solutions. First, the test conditions, including the processing chain for objective quality assessment, is explained. Then, experimental results comparing the RD performance of the proposed scalable codec using various viewpoint scalability and random access profiles are presented and discussed. Statistical results about the computational complexity and random access penalty are shown to support the experimental results analysis.

7.5.1. Test conditions

In order to evaluate the RD performance of the proposed LF coding solution the EPFL LF dataset (see Appendix A.2) is used. JPEG Pleno CTCs [35] are used to evaluate all the tested benchmarks. Several LF image coding solutions were used as benchmark, namely HEVC-PVS [64], HEVC-OPT [13], WaSP [42] and MuLE [41]. The codec HEVC-OPT, which was described in Section 7.2, is the basis of the proposed scalable codec. The proposed scalable codec is tested using two variants, the first one is HEVC-SLF, which has the scalable functionalities mentioned in Section 7.3. The second alternative is referred to as HEVC-SLF-RA and has both the scalable functionalities described in Section 7.3, as well as the viewpoint random access functionalities described in Section 7.4. Table 7.1 shows the list of tested codecs including the respective configurations. The different QPs and λ values allow the use of a common bitrate range for every tested codec, enabling a direct comparison. The HEVC-PVS uses the Low Delay with B slices configuration.

Table 7.1. List of tested codecs and respective configurations.

Codec	Configuration
HEVC-PVS HEVC-OPT HEVC-SLF HEVC-SLF-RA	$QP = [17,22,27,32,37,42]$
MuLE	$\lambda = [270,3880,30000,310000,4600000]$
WaSP	$Target\ bpp = [0.001,0.005,0.02,0.1,0.75]$

7.5.2. Viewpoint scalability performance assessment

The experimental results in Table 7.2 shows the average BD-PSNR-YUV and average BD-RATE for the twelve images of the EPFL LF dataset, comparing the proposed HEVC-SLF with HEVC-OPT, MuLE, WaSP and HEVC-PVS. From Table 7.2 it is possible to observe that the proposed HEVC-SLF outperforms all the tested benchmarks for the twelve LF images with average bitrate savings of 2.66%, 25.59%, 47.29% and 44.95%, respectively. When analyzing each result individually it is possible to see that HEVC-OPT outperforms HEVC-SLF for the LF images I09, I11 and I12 and MuLE is able to outperform HEVC-SLF for the LF images I02, in terms of bitrate savings. From the experimental results it is possible to see that the optimized RPS is able to effectively adapt to both non-scalable and scalable scanning orders, used respectively for HEVC-OPT and HEVC-SLF. The disparity estimation in both HEVC-OPT and HEVC-SFL is facilitated for lenslet LF images, because the disparity is very small. However, the proposed scalable spiral used in HEVC-SLF, in combination with the optimized RPS, creates an advantage in terms of disparity vectors statistic, when compared to HEVC-OPT using the common spiral scan . Most reference picture locations in HEVC-SLF create a disparity vector that is either horizontal or vertical, which is predicted and encoded by CABAC much more efficiently than diagonal disparity vectors. While in the case of HEVC-OPT, due to the location of the reference pictures, the disparity vectors are more likely to use horizontal and vertical components, with more combinations and larger magnitudes.

In order to further analyze the results of the proposed HEVC-SLF regarding the added viewpoint scalability features, Figure 7.9 shows the cumulative bits per layer as well as the encoding and decoding times per layer for QP 27. This QP was chosen because it represents a consistent medium point in terms of compression ratio and objective quality. Regardless, when using HEVC-SLF the lower the QP the higher the runtime and bitrate.

It is possible to see from Figure 7.9 that the number of bits per scalability layer is well distributed along the bitstream. The number of bits generated by Layer 0 is significant, knowing that only the central viewpoint is encoded. Some examples of this include, I03 and I04 where the size of Layer 0 is higher than Layer 1, which comprises eight encoded viewpoints.

As expected, Layer 5 normally carries most of the information, as it is the scalability layer with the highest number of viewpoints, i.e., 88. Additionally, it is also possible to conclude that the

compression ratio increases in layers with a higher number of encoded viewpoints. This can be observed in LFs I01 and I09, where the size per layer is similar, even though the number of viewpoints is larger. When the number of the layer increases, the disparity between viewpoints is reduced due to the scalable spiral scanning order. The last layers are composed by viewpoints that are closer to each other, which reduces the disparity, therefore the encoder is able to perform inter-view prediction more efficiently, thus increasing the compression ratio, without necessarily affecting the objective quality.

Table 7.2. BD-PSNR-YUV and BD-RATE results of the proposed HEVC-SLF vs HEVC-OPT, MULE, WASP and HEVC-PVS

Img.	vs HEVC-OPT [64]		vs MuLE [41]		vs WaSP [42]		vs HEVC-PVS [64]	
	BD-PSNR	BD-RATE	BD-PSNR	BD-RATE	BD-PSNR	BD-RATE	BD-PSNR	BD-RATE
I01	0.24 dB	-9.97 %	0.32 dB	-13.65 %	1.54 dB	-46.94 %	1.52 dB	-48.80 %
I02	0.41 dB	-16.00 %	0.00 dB	1.21 %	1.04 dB	-29.22 %	1.67 dB	-50.79 %
I03	0.22 dB	-9.11 %	0.56 dB	-18.57 %	1.17 dB	-29.81 %	1.58 dB	-48.65 %
I04	0.15 dB	-7.61 %	0.40 dB	-20.48 %	0.95 dB	-37.73 %	1.13 dB	-44.97 %
I05	0.14 dB	-7.59 %	0.60 dB	-26.05 %	1.27 dB	-42.75%	1.15 dB	-48.33 %
I06	0.06 dB	-4.82 %	1.44 dB	-52.35 %	2.45 dB	-73.46 %	0.93 dB	-43.51 %
I07	0.05 dB	-2.00 %	0.59 dB	-19.78 %	1.68 dB	-43.61 %	1.03 dB	-39.61 %
I08	0.06 dB	-4.04 %	1.48 dB	-50.51 %	2.26 dB	-66.43 %	0.93 dB	-40.71 %
I09	-0.12 dB	6.10 %	0.65 dB	-21.31 %	1.30 dB	-32.92 %	1.35 dB	-45.94 %
I10	0.01 dB	-0.88 %	0.62 dB	-24.62 %	1.37 dB	-46.33 %	1.26 dB	-45.69 %
I11	-0.22 dB	14.60 %	0.59 dB	-22.51 %	2.61 dB	-66.20 %	0.71 dB	-35.57 %
I12	-0.16 dB	7.64 %	1.30 dB	-38.51 %	2.21 dB	-52.11 %	1.45 dB	-46.85 %
Avg.	0.17 dB	-2.66 %	0.71 dB	-25.59 %	1.65 dB	-47.29 %	1.23 dB	-44.95 %

The computational complexity, in terms of runtime, of the proposed HEVC-SLF is compared to its counterparts in Figure 7.9, for every scalability layer for images encoded with QP 27. Additionally, the computational complexity of the tested benchmarks for I04 is shown in Table 7.3. These tests were performed using a PC equipped with an Intel Core i7 CPU 4790K@4.0GHz and 32GB of RAM, running Ubuntu 16.04 The runtimes for MuLE and WaSP were obtained for cases where the resultant objective quality is similar to the HEVC-based codecs, i.e., MuLE using a λ of 3880 and WaSP using a target bitrate of 0.1 bpp. From Table 7.3 it is possible to observe that all the HEVC-based encoder and decoder have similar runtimes. MuLE and WaSP have faster encoders than the HEVC-based codecs, however, their decoders are slower than the HEVC-based decoders. From Figure 7.9 it is also possible to observe that, differently from the distribution of bits per scalability layer, the time required to encode and

decode each viewpoint is similar. Consequently, the time required to encode and decode each scalability layer is roughly proportional to the number of viewpoints in each layer. The only exception is Layer 0, that is encode as an intra frame, which takes less time to encode than the inter frames, but it takes longer runtime to decode. It is also worth mentioning that Layer 0 encoding time is very small relative to the remaining layers, which is the reason why it is not noticeable in Figure 7.9. The steady behavior of the computational complexity codec along the several scalability layers, as mentioned in Section 7.3, may be advantageous in scenarios where the existing computational power is scarce.

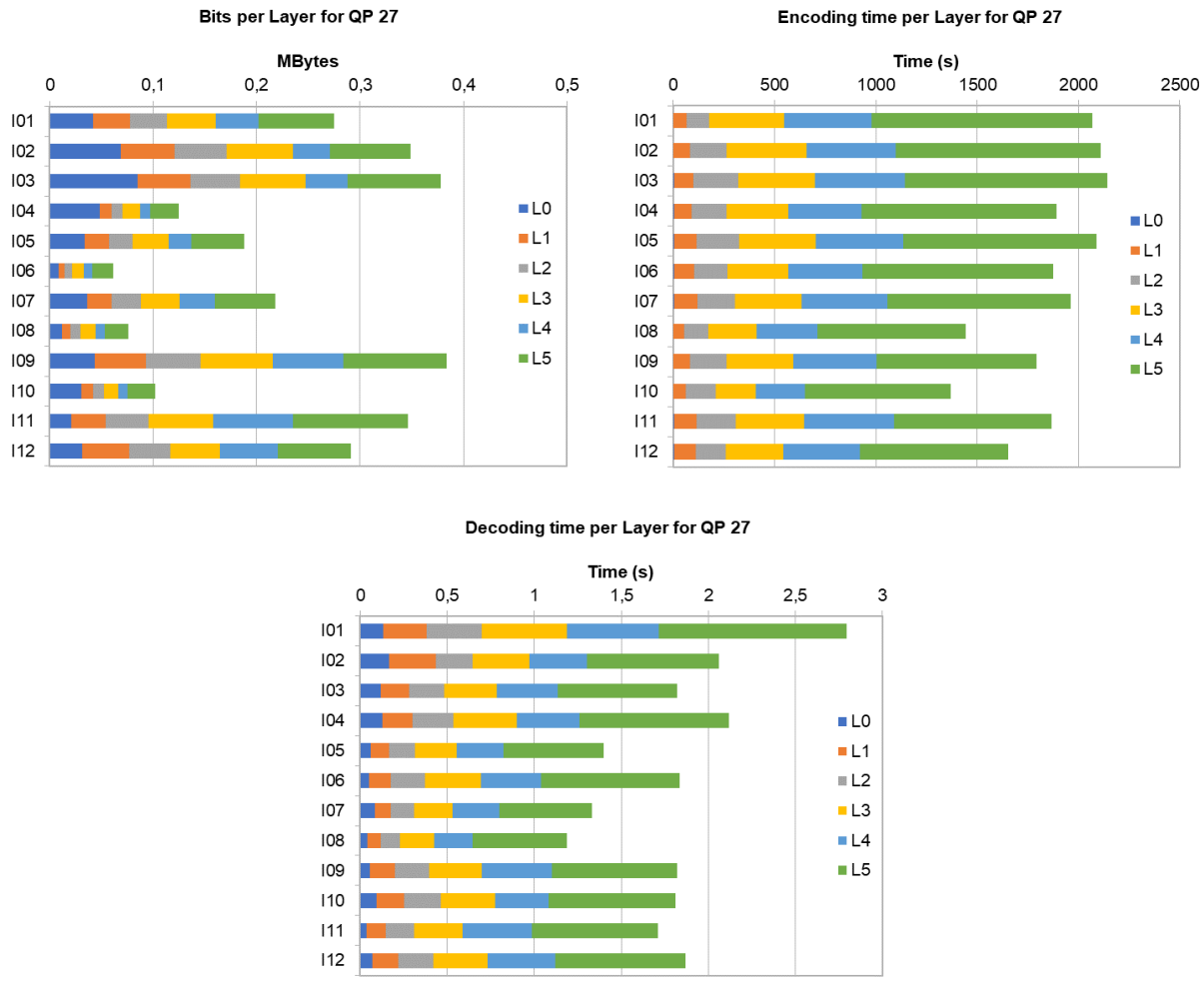


Figure 7.9. Statistical coding information of HEVC-SLF, including bits per layer (top-left), encoding time per layer (top-right) and decoding time per layer (bottom) for QP 27.

Table 7.3. Computational complexity comparison of the tested benchmarks.

Codec	Encoder		Decoder	
	Runtime	vs HEVC-PVS	Runtime	vs HEVC-PVS
HEVC-PVS	1192 s	-	2.83 s	-
HEVC-OPT	1018 s	0.85	**2.96 s**	**1.05**
MuLE	**209 s**	**0.18**	15.67 s	5.54
WaSP*	214 s	0.18	32.68 s	11.55
HEVC-SLF	1245 s	1.04	3.83 s	1.35

*using multithread (8 threads)

7.5.3. Viewpoint Random access performance assessment

The experimental results of comparing viewpoint random access for several combinations of coding profiles are shown in Table 7.4 and Table 7.5. All the three proposed control parameters described in Section 7.4 were used to test the proposed HEVC-SLF-RA. These control parameters include the RPL size per viewpoint, i.e., two or four; the maximum scalability layer that can contain reference pictures, i.e., six (all), four, three and two; and, the viewpoint region and the number of intra frames, which can be categorized as the following scenarios: 1Region-1Intra, 2Region-1Intra, 4Region-1Intra, 5Region-5Intra and 9Region-9Intra.

The values marked in bold in Table 7.4 and Table 7.5 correspond to the coding profiles that achieve higher coding efficiency than HEVC-PVS. The average, standard deviation, maximum and minimum RAP values were calculated for the twelve LF images used for testing (encoded with a QP 27). Note that, HEVC-SLF corresponds to the following coding profile: Max. reference per viewpoint of four, 1Region-1Intra, six reference scalability layers. As previously mentioned, QP 27 was chosen because it represents a consistent medium point in terms of compression ratio and objective quality. It was also verified that the RAP will decrease for higher QPs, i.e., lower objective quality.

From the results in Table 7.4 and Table 7.5 it is possible to observe that the proposed combination of control parameters allows for a vast number of coding profiles that balance the tradeoff between coding efficiency and the viewpoint random access capabilities. Such coding profiles used to restrict the inter-viewpoint dependencies, consequently, decrease the RAP for the HEVC-SLF-RA codec. However, for some coding profiles, namely when testing an RPL size per viewpoint of two (Table 7.4), the results are identical when using 1Region-1Intra,

2Region-1Intra and 4Region-1Intra. This occurs because the inter-viewpoint dependencies are only affected after the third reference picture. However, since in this case the RPL size is two, the reference picture selection is the same and, therefore, the results for RAP and coding efficiency are unchanged. Figure 7.10 shows different plots of the same points as in Table 7.4 and Table 7.5, comparing the maximum RAP and the AVG. BD-RATE against HEVC-PVS.

Table 7.4. HEVC-SLF-RA random access penalty and average BD-RATE vs HEVC-PVS for the different profile combinations for twelve LF images when using RPL size per viewpoint of two

RPL size per viewpoint = 2						
Regions/ Intra	Ref. Layers	AVG.	STD	MAX	MIN	AVG. BD-RATE
1/1	6	**0.2395**	**0.0843**	**0.4914**	**0.0496**	**-36.44 %**
	4	**0.2366**	**0.0834**	**0.4730**	**0.0491**	**-35.65 %**
	3	**0.1825**	**0.0671**	**0.3731**	**0.0413**	**-17.59 %**
	2	0.1444	0.0557	0.3121	0.0355	2.69 %
2/1	6	**0.2395**	**0.0843**	**0.4914**	**0.0496**	**-36.44 %**
	4	**0.2366**	**0.0834**	**0.4730**	**0.0491**	**-35.65 %**
	3	**0.1825**	**0.0671**	**0.3731**	**0.0413**	**-17.59 %**
	2	0.1444	0.0557	0.3121	0.0355	2.69 %
4/1	6	**0.2395**	**0.0843**	**0.4914**	**0.0496**	**-36.45 %**
	4	**0.2366**	**0.0834**	**0.4730**	**0.0491**	**-35.65 %**
	3	**0.1825**	**0.0671**	**0.3731**	**0.0413**	**-17.59 %**
	2	0.1444	0.0557	0.3121	0.0355	2.69 %
5/5	6	**0.0975**	**0.0275**	**0.2244**	**0.0181**	**-0.01 %**
	4	0.0948	0.0272	0.2226	0.0176	2.30 %
	3	0.0771	0.0239	0.1919	0.0149	25.05 %
	2	0.0596	0.0207	0.1633	0.0129	52.78 %
9/9	6	0.0723	0.0213	0.1676	0.0174	15.51 %
	4	0.0694	0.0213	0.1657	0.0163	20.10 %
	3	0.0569	0.0204	0.1487	0.0140	42.71 %
	2	0.0468	0.0181	0.1350	0.0125	62.13 %

The coding efficiency can be analyzed from three different perspectives:

- **RPL size**: From the plot on the left of Figure 7.10, it is possible to see that when using a lower RPL size per viewpoint, a reduction between 0% and 10% of bitrate savings is expected, which corresponds to a reduction between 0 and 0.1 of RAP.
- **Viewpoint region**: From the middle plot in Figure 7.10, the consequences of the different viewpoint regions are evaluated, where it is possible to see that the number of intra frames creates well defined curves, i.e., the first three configurations, which use one intra (grey, orange and dark blue), five intra (yellow) and nine intra (light blue)

configuration. There is little variation when changing the number of viewpoint regions, without changing the number of intra frames. Additionally, when considering bitrate savings of roughly 5% over HEVC-PVS it is possible to observe that the 5Region-5Intra option is superior to a single intra frame in terms of RAP.

- **Maximum scalability layer**: From the right plot in Figure 7.10 it is possible to see that the reduction in the number of reference scalability layers only becomes more evident when using less than four reference scalability layers. In most cases the difference in terms of coding efficiency and RAP is low when using six or four reference scalability layers.

Table 7.5. HEVC-SLF-RA random access penalty and average BD-RATE vs HEVC-PVS for the different profile combinations for twelve LF images when using RPL size per viewpoint of four

RPL size per viewpoint = 4						
Regions/ intra	Ref. Layers	AVG.	STD	MAX	MIN	AVG. BD-RATE
1/1	6	**0.3583**	**0.0898**	**0.5875**	**0.0599**	**-44.95%**
	4	**0.3220**	**0.0912**	**0.5474**	**0.0537**	**-42.15 %**
	3	**0.2318**	**0.0685**	**0.4160**	**0.0444**	**-23.76 %**
	2	**0.1551**	**0.0560**	**0.3204**	**0.0366**	**-0.65 %**
2/1	6	**0.3452**	**0.0874**	**0.5736**	**0.0597**	**-44.57 %**
	4	**0.3090**	**0.0886**	**0.5304**	**0.0537**	**-41.8 %**
	3	**0.2214**	**0.0680**	**0.4089**	**0.0440**	**-23.2 %**
	2	**0.1539**	**0.0552**	**0.3163**	**0.0365**	**-0.09 %**
4/1	6	**0.3370**	**0.0873**	**0.5635**	**0.0590**	**-44.39 %**
	4	**0.3006**	**0.0884**	**0.5213**	**0.0531**	**-41.61 %**
	3	**0.2150**	**0.0683**	**0.4070**	**0.0437**	**-22.81 %**
	2	0.1535	0.0550	0.3151	0.0364	0.43 %
5/5	6	**0.1276**	**0.0298**	**0.2342**	**0.0205**	**-6.81 %**
	4	**0.1107**	**0.0275**	**0.2269**	**0.0182**	**-0.85 %**
	3	0.0856	0.2420	0.1986	0.0154	21.50 %
	2	0.0596	0.0207	0.1633	0.0129	52.78 %
9/9	6	0.088	0.0230	0.1729	0.0192	9.84 %
	4	0.0775	0.0232	0.1691	0.0169	16.71 %
	3	0.0603	0.0219	0.1520	0.0141	40.89 %
	2	0.0468	0.0181	0.1350	0.0125	62.13 %

Figure 7.10. Tradeoff between the Maximum RAP for twelve images encoded with QP 27 and the average BD-RATE savings against HEVC-PVS.

From the vast number of coding profiles that generate different tradeoffs between random access capabilities and coding efficiency, the authors created a list of suggested profiles based on four different tradeoff points. This list is shown in Table 7.6, which includes a maximum coding efficiency profile, achieving more than 40% bitrate savings over HEVC-PVS; two balanced profile biased towards coding efficiency and random access capabilities, achieving more than 20% and 5% bitrate savings over HEVC-PVS, respectively; and, a maximum viewpoint random access capabilities profile that values the RAP over the coding efficiency, achieving a maximum RAP of 0.17. Although the "Max. RA" profile is 20% less efficient than HEVC-PVS, the maximum RAP is much superior, considering that HEVC-PVS maximum RAP is 1.

Table 7.6. List of four suggested profiles with different tradeoffs between the random access capabilities and coding efficiency.

Suggested Profiles	Max. Ref	Ref. Layers	Regions/ Intra	Max. RAP	BD-RATE.
Max. Eff.	4	6	4/1	0.5635	-44.39%
Balanced High Eff.	4	3	4/1	0.4070	-22.81%
Balanced High RA	4	6	5/5	0.2342	-6.81 %
Max. RA	2	4	9/9	0.1657	20.10%

7.6. Final remarks

In this chapter, a new coding approach that allows viewpoint scalability and random access capabilities for LF is proposed. The availability of these features, while maximizing the coding efficiency, can be achieved by using a new optimized RPS method that is able to adapt to any scanning order. This technique is very important to accommodate the new scanning order proposed for the viewpoint scalability features, allowing to maintain the coding efficiency comparable to the non-scalable version. Additionally, the proposed control parameters allow to exploit a vast number of coding profiles, enabling a fine control over the dependencies between viewpoints, therefore, the viewpoint random access capabilities.

When compared to the state-of-the-art HEVC-PVS, the proposed HEVC-SLF codec achieved, for the EPFL LF dataset, an average bitrate saving of 44.95%, which is 2.66% more efficient than the its non-scalable version (HEVC-OPT). In comparison to the benchmarks MuLE and WaSP, which are being considered for the JPEG Pleno standard, the proposed HEVC-SLF also achieves average bitrate savings of 25.59% and 47.29%, respectively.

To enable viewpoint random access capabilities, a scalable codec HEVC-SLF-RA was proposed, introducing a vast range of random access profiles. According to the application, these profiles can be used to control the tradeoff between the random access capabilities and the coding efficiency. By acting on the control parameters, the maximum RAP of the proposed HEVC-SLF-RA in relation HEVC-PVS (RAP=1) ranges from 0.22 to 0.56, respectively, for bit rating savings from 0 to 44.95%.

Chapter 8. Achievements and future work

8.1. Achievements

In this Thesis, several contributions have been proposed towards advancing the state-of-the-art on LF image representation and coding. The work developed in the context of this Thesis involved the research and development of techniques related to several steps of the LF transmission/storage pipeline in order to develop efficient LF coding solutions.

The first achievement of this Thesis concerns the in-depth analysis of the most popular state-of-the-art LF representations. This analysis, performed in Chapter 4, compares the coding efficiency of the most popular LF representations, namely, LL and 4D LF, including both MI and SAI-based variants. HEVC has been selected as the coding solution for this analysis since it provides powerful intra and inter prediction tools able to exploit the different types of redundancy inherent in LF content. The achieved coding performance by HEVC is a useful benchmark to establish the baseline results for what is expected from each LF representation when a non-specific LF coding solution is used to encode LF content. As a result of this analysis, it was concluded that the 4DLF-PVS representation achieves the highest coding efficiency. Moreover, it was also concluded that both LL and MI-based variants of 4D LF representations tend to be less efficient than SAI-based variants since there are no prediction tools are available in HEVC to exploit the MI redundancy.

The second achievement of this Thesis concerns the development of efficient coding solutions that support both LL and 4D LF representations. Both data representations were exhaustively tested and evaluated in Chapter 4 in terms of coding efficiency. When comparing the MI-based variants in Chapter 4, it was concluded that the LL representation, although more prone to color distortion, achieved higher coding efficiency. This result is important as it is expected that some

applications may benefit from the more compact nature of LL. Therefore, Chapter 5 proposed a coding solution, which is able to encode any MI-based LF image, regardless of the camera type and data representation (LL or 4DLF-MI). In this context, two algorithms were proposed based on the HOP model, which allow for up to eight DoF when exploiting the inter-MI redundancy. The first algorithm uses the HOP model to estimate the GT that best approximates a target block within a reference region containing neighboring MIs, explicitly transmitting the HOP information in the form of one translational vector and three (six DoF) to four (eight DoF) HOP vectors. The second algorithm uses the HOP model to estimate the GT through a training step that optimizes the match between several predetermined pairs of two quadrilaterals of the reference region, i.e., several training directions. This allows the HOP information to be implicitly available at the decoder side, by repeating the selected training step, providing a higher RD efficiency. Both algorithms were able to consistently outperform state-of-the-art solutions based on LOP models, for both LL and 4D LF representations.

The third achievement of this Thesis concerns the effective exploitation of MI and SAI redundancies. As introduced in Chapter 3, there are four types of redundancies that stem from two types of representations, MIs and SAIs. While Chapter 5 proposed two solutions to exploit the Inter-MI redundancy, which are based on the HOP model, Chapter 6 proposed a LF image codec based on a hybrid representation, which uses both 4DLF-MI and 4DLF-PVS representations. The proposed codec allows both MI and SAI redundancies to be exploited, since the encoder is able to select the type of representation (and associated prediction mode) that achieves the highest RD performance. When the MI based representation is used, several prediction modes are available to exploit the intra-MI redundancy, such as DC, MED, GAP and AGSP. To exploit the inter-MI redundancy, several LSP-based prediction modes were developed specifically for LF data. Experimental results show that the proposed codec, based on this proposed hybrid representation, is more efficient than the versions that rely exclusively on MIs or SAIs. Finally, Chapter 7 proposed a solution that takes advantage of the HEVC intra and inter prediction tools to exploit the intra-SAI and inter-SAI redundancy, respectively. This solution signals the scanning order used to generate the 4DLF-PVS signal, which allows the encoder and decoder to implicitly optimize the RPLs based on a distance minimization between each encoded SAI and the current SAI.

The fourth achievement of this Thesis concerns the support for viewpoint scalability and random access. The LF image codec based on the 4DLF-PVS representation proposed in Chapter 7 was modified to provide additional functionalities like viewpoint scalability and random access. A novel scalable spiral scan was proposed, allowing for six layers of viewpoint scalability. The gradual increase in viewpoints in each scalability layer effectively allows for compatibility with legacy 2D and 3D/stereoscopic displays, as well as with LF displays with different capabilities in terms of angular resolutions. Additionally, it was demonstrated the comparable efficiency between both scalable and non-scalable variants, since the proposed LF image codec adapts to the scalable scanning order. This scalable codec also allows for several configurations that alter the inter-dependencies between SAIs, which has the advantage of allowing for viewpoint random access. Control parameters that limit the RPS in terms of access to the different scalability layers and LF regions allow a fine adjustment of the tradeoff between viewpoint random access and coding efficiency. Maximizing the viewpoint random access capabilities, or even a more balanced configuration, that, while penalizing coding efficiency, allows for improved LF navigation efficiency, as well as lower decoding delay and lower computational complexity.

8.2. Future Work

The development of the contributions described in the previous section also raised some possible lines for future work to further improve the LF coding efficiency and functionalities, related with:

Improved HOP approach

The main results produced by the proposed HOP model in Chapter 5 showed potential when applied to LF image coding. However, the coding efficiency of the HOP model can be further improved by performing efficient HOP vector prediction. This would require a study on the HOP vector statistics for several LF images and block sizes. It is expected that in the case of the LF images captured using a focused LF camera, the combination of vectors would be correlated with the disparity between neighboring MIs. This correlation would produce good vector predictions and potentially decrease the bitrate required to transmit the HOP information. Additionally, the computational complexity can potentially be reduced by having an effective

HOP vector prediction, since, if the HOP vectors can be accurately predicted, a reduced number of vector combinations needs to be tested.

Improved coding efficiency for focused LF cameras

With the exception of Chapter 5, which was developed for both UNF and FOC LF cameras, most of the proposed approaches were proposed specifically considering the UNF LF camera, because of its simplicity when it comes to viewpoint rendering. However, the FOC LF camera, despite having typically a lower angular resolution, generates viewpoints with higher spatial resolution than an UNF LF camera. This characteristic may be important for some applications, therefore, future work may include developing techniques exclusively for FOC LF cameras by optimizing the RD criteria based on the rendered viewpoints. Alternatively, the approach proposed in Chapter 6 can be adapted to better accommodate FOC LF images by altering the conversion algorithm that is applied between the MI and SAI representation. In the proposed codec, since it was specifically designed for the UNF LF cameras, the conversion is applied with a 1×1 patch since each pixel in each MI is part of a different SAI, however when encoding FOC LF images, different patch sizes could be tested for improved coding efficiency.

Extend techniques to HDCA signals

A generic solution for LFs captured with both HDCAs and MLA-based LF cameras is also in line with the standardization initiatives goals, as it would allow a single solution to be used to encode content with varying degrees of angular resolutions and camera baselines. The proposed techniques in Chapter 6 and 7 are based on the 4D LF data representation, which allows for their application in LF images generated from both UNF LF cameras and HDCAs. It is expected that the application of such techniques to HDCA signals can be performed almost in a seamless way, i.e., with minimum changes to the current implementations. Nevertheless, since the main difference between both capturing devices is the baseline, several techniques can be proposed that would be able to adapt to both narrow and wide baselines. Additionally, the scalability and random access functionalities allowed by Chapter 7 would also benefit end-user when used to transmit/store HDCA signals.

Appendix A. Light field test content

This appendix illustrates the LF test content that was used to validate the several contributions throughout this Thesis. Section A.1. presents the FOC LF test content and Section A.2. presents the UNF LF test content.

A.1. Focused

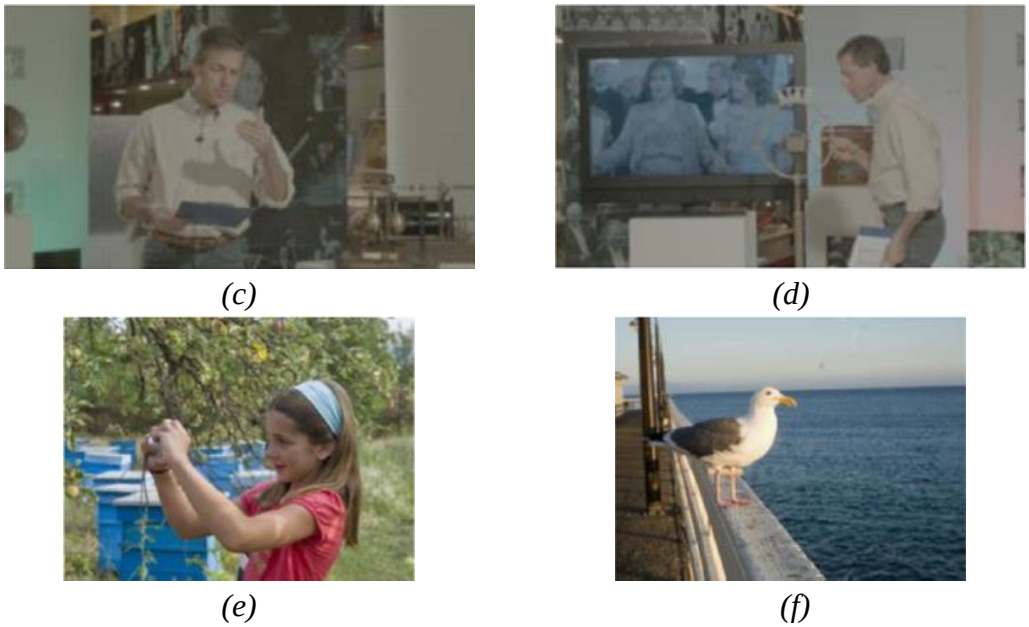
The FOC LF test content that was used in this Thesis includes images from project 3D-VIVANT dataset [125] and from Todor Georgiev LF dataset [126]. These LF images are listed in Table A.1 and the central rendered view of each corresponding LF image is shown in Figure A.1.

Table A.1. Main characteristics of the FOC LF test content

Name (frame)	Figure	Dataset	Resolution	MI resolution	MLA type
Plane and Toy (0)	(a)	3D-VIVANT [125]	1920×1088	28×28	Square grid / Square MIs
Plane and Toy (150)	(b)				
Demichelis Spark (0)	(c)		2880×1620	38×38	Square grid / Circular MIs
Demichelis Cut (0)	(d)				
Laura	(e)	T. Georgiev [126]	7240×5432	75×75	Square grid / Square MIs
Seagull	(f)				

(a)

(b)



(c) *(d)*

(e) *(f)*

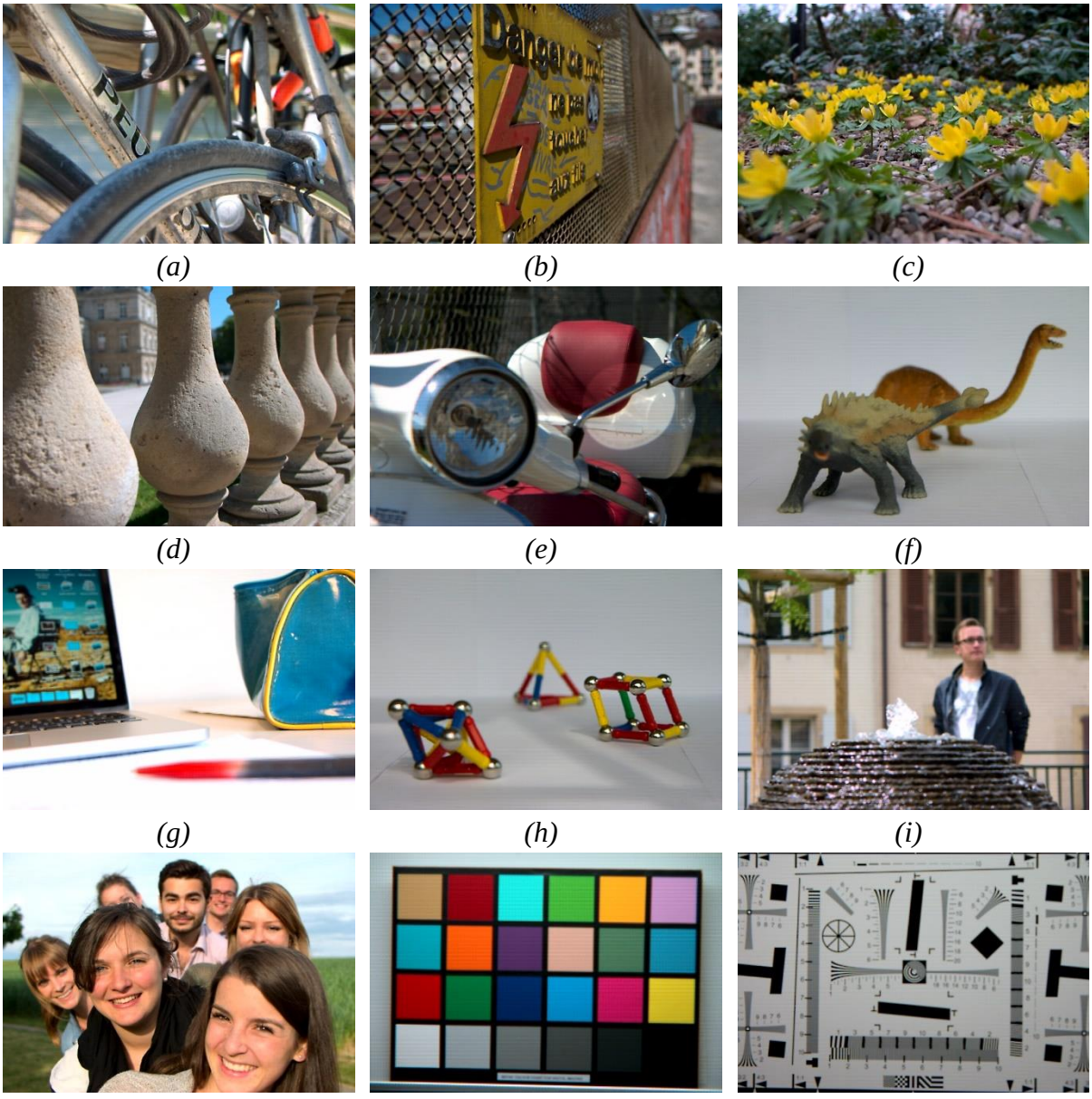
Figure A.1. Rendered central view from each FOC LF image.

A.2. Unfocused

The UNF LF test content that was used in this Thesis includes images the EPFL LF dataset [22]. These LF images are listed in Table A.2 and the central rendered view of each corresponding LF image is shown in Figure A.2.

Table A.2. Main characteristics of the UNF LF test content

Name	Code	Figure	Dataset	Resolution	MI resolution	MLA type
Bikes	I01	(a)	EPFL [22]	7728×5368	15×15	Hexagonal grid / Circular MIs
Danger de mort	I02	(b)				
Flowers	I03	(c)				
Stone pillars outside	I04	(d)				
Vespa	I05	(e)				
Ankylosaurus & Diplodocus 1	I06	(f)				
Desktop	I07	(g)				
Magnets 1	I08	(h)				
Fountain & Vincent 2	I09	(i)				
Friends 1	I10	(j)				
Color chart 1	I11	(k)				
ISO chart 1	I12	(l)				



(a) *(b)* *(c)*

(d) *(e)* *(f)*

(g) *(h)* *(i)*

(j) *(k)* *(l)*

Figure A.2. Rendered central view from each UNF LF image.

References

[1] *Cisco Visual Networking Index: Forecast and Trends, 2017–2022 White Paper*. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html.

[2] *ITU-T IMT-2020,* https://www.itu.int/en/ITU-R/study-groups/rsg5/rwp5d/imt-2020/Pages/default.aspx.

[3] B. Bellalta, "IEEE 802.11ax: High-efficiency WLANS," *IEEE Wirel. Commun.*, vol. 23, no. 1, pp. 38–46, Feb. 2016, doi: 10.1109/MWC.2016.7422404.

[4] T. Georgiev and A. Lumsdaine, "Rich Image Capture with Plenoptic Cameras," in *IEEE International Conference on Computational Photography*, Cluj-Napoca, Romania, 2010, pp. 1–8.

[5] C. Hahne, A. Aggoun, S. Haxha, V. Velisavljevic, and J. C. J. Fernández, "Light field geometry of a standard plenoptic camera," *Opt Express*, vol. 22, no. 22, pp. 26659–26673, Nov. 2014, doi: 10.1364/OE.22.026659.

[6] A. Lumsdaine and T. Georgiev, "The focused plenoptic camera," in *IEEE International Conference on Computational Photography*, 2009, pp. 1–8, doi: 10.1109/ICCPHOT.2009.5559008.

[7] "JPEG PLENO Abstract and Executive Summary," Sydney, ISO/IEC JTC 1/SC 29/WG1 N6922, Feb. 2015.

[8] "MPEG-I Technical Report on Immersive Media," Torino, Italy, ISO/IEC JTC1/SC29/WG11 N17069, Jul. 2017.

[9] R. J. S. Monteiro, N. M. M. Rodrigues, S. M. M. Faria, and P. J. L. Nunes, "Light field image coding: objective performance assessment of Lenslet and 4D LF data representations," in *Proc.SPIE*, 2018, vol. 10752, pp. 10752-10752–17, doi: 10.1117/12.2322713.

[10] R. Monteiro *et al.*, "Light field HEVC-based image coding using locally linear embedding and self-similarity compensated prediction," in *IEEE International Conference on Multimedia Expo Workshops*, 2016, pp. 1–4, doi: 10.1109/ICMEW.2016.7574670.

[11] R. J. Monteiro, P. Nunes, N. Rodrigues, and S. M. M. de Faria, "Light Field Image Coding using High Order Intra Block Prediction," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 7, pp. 1120–1131, Oct. 2017, doi: 10.1109/JSTSP.2017.2721358.

[12] R. J. S. Monteiro, P. J. L. Nunes, S. M. M. Faria, and N. M. M. Rodrigues, "Light Field Image Coding using High Order Prediction Training," in *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 1845–1849, doi: 10.23919/EUSIPCO.2018.8553150.

[13] R. J. S. Monteiro, P. J. L. Nunes, S. M. M. Faria, and N. M. M. Rodrigues, "Optimized Reference Picture Selection for Light Field Image Coding," in *2019 27th European Signal Processing Conference (EUSIPCO)*, 2019, pp. 1–4.

[14] D. G. Dansereau, O. Pizarro, and S. B. Williams, "Decoding, Calibration and Rectification for Lenselet-Based Plenoptic Cameras," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1027–1034, doi: 10.1109/CVPR.2013.137.

[15] M. Levoy, "Light Fields and Computational Imaging," *Computer*, vol. 39, no. 8, pp. 46–55, Aug. 2006, doi: 10.1109/MC.2006.270.

[16] *The Stanford Multi-Camera Array*. http://graphics.stanford.edu/projects/array/.

[17] *Fraunhofer IIS Light-field Technology*. https://www.iis.fraunhofer.de/en/ff/amm/for/forschbewegtbildtechn/lichtfeld.html.

[18] M. Levoy *et al.*, "The Digital Michelangelo Project: 3D Scanning of Large Statues," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, New York, NY, USA, 2000, pp. 131–144, doi: 10.1145/344779.344849.

[19] G. Wu *et al.*, "Light Field Image Processing: An Overview," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 7, pp. 926–954, Oct. 2017, doi: 10.1109/JSTSP.2017.2747126.

[20] *Lytro Illum*. http://lightfield-forum.com/2014/04/lytro-illum-tech-specs-and-details-what-you-need-to-know/.

[21] *Raytrix R42*. http://lightfield-forum.com/2014/04/lytro-illum-tech-specs-and-details-what-you-need-to-know/.

[22] *EPFL Light-field image dataset*. http://mmspg.epfl.ch/EPFL-light-field-image-dataset.

[23] *MPEG-I ISO/IEC 23090*. https://mpeg.chiariglione.org/standards/mpeg-i.

[24] P. Schelkens *et al.*, "JPEG Pleno: Providing representation interoperability for holographic applications and devices," *ETRI J.*, vol. 41, no. 1, pp. 93–108, 2019, doi: 10.4218/etrij.2018-0509.

[25] "MPEG-I Phase 1 Use Cases," Ljubljana, Slovenia, ISO/IEC JTC1/SC29/WG11/N17886, Jul. 2018.

[26] "Collection of usage scenarios and current statuses of advanced immersive audio-visual systems," Report ITU-R BT.2420-0, Apr. 2018.

[27] R. Ng, "Digital Light Field Photography," PhD Thesis, Stanford University, Stanford, CA, USA, 2006.

[28] S. Zhang, H. Sheng, C. Li, J. Zhang, and Z. Xiong, "Robust depth estimation for light field via spinning parallelogram operator," *Comput. Vis. Image Underst.*, vol. 145, pp. 148–159, 2016, doi: https://doi.org/10.1016/j.cviu.2015.12.007.

[29] *FOVI3D LF Display*. http://www.fovi3d.com/.

[30] *Holografika*. https://holografika.com/.

[31] *Oculus Quest*. https://www.oculus.com/quest/.

[32] *Valve Index*. https://www.valvesoftware.com/pt/index.

[33] D. Lanman and D. Luebke, "Near-eye Light Field Displays," *ACM Trans Graph*, vol. 32, no. 6, pp. 220:1–220:10, Nov. 2013, doi: 10.1145/2508363.2508366.

[34] *Light Field Toolbox v0.4*. http://dgd.vision/Tools/LFToolbox/.

[35] "JPEG Pleno Light Field Coding Common Test Conditions," Geneva, Switzerland, ISO /IEC JTC 1/SC 29 /WG 1 N83029, Mar. 2019.

[36] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012, doi: 10.1109/TCSVT.2012.2221191.

[37] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003, doi: 10.1109/TCSVT.2003.815165.

[38] A. Norkin *et al.*, "HEVC Deblocking Filter," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1746–1754, Dec. 2012, doi: 10.1109/TCSVT.2012.2223053.

[39] C. Fu *et al.*, "Sample Adaptive Offset in the HEVC Standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1755–1764, Dec. 2012, doi: 10.1109/TCSVT.2012.2221529.

[40] "JPEG Pleno Plenoptic image coding system - Part 2: Light Field Coding," ISO/IEC JTC1/SC 29 WG1N84047, Aug. 2019.

[41] M. B. de Carvalho *et al.*, "A 4D DCT-Based Lenslet Light Field Codec," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 435–439, doi: 10.1109/ICIP.2018.8451684.

[42] P. Astola and I. Tabus, "WaSP: Hierarchical Warping, Merging, and Sparse Prediction for Light Field Image Compression," in *2018 7th European Workshop on Visual Information Processing (EUVIP)*, 2018, pp. 1–6, doi: 10.1109/EUVIP.2018.8611756.

[43] P. Helin, P. Astola, B. Rao, and I. Tabus, "Minimum Description Length Sparse Modeling and Region Merging for Lossless Plenoptic Image Compression," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 7, pp. 1146–1161, Oct. 2017, doi: 10.1109/JSTSP.2017.2737967.

[44] C. Conti, J. Lino, P. Nunes, L. D. Soares, and P. L. Correia, "Spatial prediction based on self-similarity compensation for 3D holoscopic image and video coding," in *IEEE International Conference on Image Processing*, Brussels, Belgium, 2011, pp. 961–964, doi: 10.1109/ICIP.2011.6116721.

[45] C. Conti, J. Lino, P. Nunes, and L. D. Soares, "Spatial and temporal prediction scheme for 3D holoscopic video coding based on H.264/AVC," in *International Packet Video Workshop*, 2012, pp. 143–148, doi: 10.1109/PV.2012.6229727.

[46] C. Conti, P. Nunes, and L. D. Soares, "New HEVC prediction modes for 3D holoscopic video coding," in *IEEE International Conference on Image Processing*, 2012, pp. 1325–1328, doi: 10.1109/ICIP.2012.6467112.

[47] Y. Li, M. Sjöström, R. Olsson, and U. Jennehag, "Efficient intra prediction scheme for light field image compression," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2014, pp. 539–543, doi: 10.1109/ICASSP.2014.6853654.

[48] L. F. R. Lucas *et al.*, “Locally linear embedding-based prediction for 3D holoscopic image coding using HEVC,” in *European Signal Processing Conference*, Lisbon, Portugal, 2014, pp. 11–15.

[49] D. Liu, P. An, R. Ma, and L. Shen, “Disparity compensation based 3D holoscopic image coding using HEVC,” in *IEEE China Summit and International Conference on Signal and Information Processing*, 2015, pp. 201–205, doi: 10.1109/ChinaSIP.2015.7230391.

[50] C. Conti, L. D. Soares, and P. Nunes, “HEVC-based 3D holoscopic video coding using self-similarity compensated prediction,” *Signal Process. Image Commun.*, vol. 42, pp. 59–78, Mar. 2016, doi: 10.1016/j.image.2016.01.008.

[51] C. Conti, P. Nunes, and L. D. Soares, “HEVC-based light field image coding with bi-predicted self-similarity compensation,” in *IEEE International Conference on Multimedia Expo Workshops*, 2016, pp. 1–4, doi: 10.1109/ICMEW.2016.7574667.

[52] Y. Li, R. Olsson, and M. Sjöström, “Compression of unfocused plenoptic images using a displacement intra prediction,” in *IEEE International Conference on Multimedia Expo Workshops*, 2016, pp. 1–4, doi: 10.1109/ICMEW.2016.7574673.

[53] Y. Li, M. Sjöström, R. Olsson, and U. Jennehag, “Coding of Focused Plenoptic Contents by Displacement Intra Prediction,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 7, pp. 1308–1319, Jul. 2016, doi: 10.1109/TCSVT.2015.2450333.

[54] D. Liu, P. An, R. Ma, C. Yang, and L. Shen, “3D holoscopic image coding scheme using HEVC with Gaussian process regression,” *Signal Process. Image Commun.*, vol. 47, pp. 438–451, 2016, doi: https://doi.org/10.1016/j.image.2016.08.004.

[55] C. Conti, P. Nunes, and L. D. Soares, “Light field image coding with jointly estimated self-similarity bi-prediction,” *Signal Process. Image Commun.*, vol. 60, pp. 144–159, Feb. 2018, doi: https://doi.org/10.1016/j.image.2017.10.006.

[56] D. Liu, P. An, R. Ma, W. Zhan, X. Huang, and A. A. Yahya, “Content-based Light Field Image Compression Method with Gaussian Process Regression,” *IEEE Trans. Multimed.*, pp. 1–1, 2019, doi: 10.1109/TMM.2019.2934426.

[57] M. C. Forman and A. Aggoun, “Quantisation strategies for 3D-DCT-based compression of full parallax 3D images,” in *International Conference on Image Processing and Its Applications*, 1997, vol. 1, pp. 32–35, doi: 10.1049/cp:19970848.

[58] A. Aggoun, "A 3D DCT Compression Algorithm For Omnidirectional Integral Images," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2006, vol. 2, pp. II–II, doi: 10.1109/ICASSP.2006.1660393.

[59] H. Han, X. Jin, and Q. Dai, "Lenslet image compression based on image reshaping and macro-pixel Intra prediction," in *2017 IEEE International Conference on Multimedia and Expo (ICME)*, 2017, pp. 1177–1182, doi: 10.1109/ICME.2017.8019424.

[60] R. Zhong, I. Schiopu, B. Cornelis, S. Lu, J. Yuan, and A. Munteanu, "Dictionary Learning-based, Directional and Optimized Prediction for Lenslet Image Coding," *IEEE Trans. Circuits Syst. Video Technol.*, pp. 1–1, 2018, doi: 10.1109/TCSVT.2018.2826052.

[61] R. Zhong, S. Wang, B. Cornelis, Y. Zheng, J. Yuan, and A. Munteanu, "L1-optimized linear prediction for light field image compression," in *2016 Picture Coding Symposium (PCS)*, 2016, pp. 1–5, doi: 10.1109/PCS.2016.7906404.

[62] T. Zhong, X. Jin, L. Li, and Q. Dai, "Light Field Image Compression Using Depth-based CNN in Intra Prediction," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 8564–8567, doi: 10.1109/ICASSP.2019.8682820.

[63] G. Tech, Y. Chen, K. Müller, J. Ohm, A. Vetro, and Y. 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, doi: 10.1109/TCSVT.2015.2477935.

[64] A. Vieira, H. Duarte, C. Perra, L. Tavora, and P. Assuncao, "Data formats for high efficiency coding of Lytro-Illum light fields," in *International Conference on Image Processing Theory, Tools and Applications*, 2015, pp. 494–497, doi: 10.1109/IPTA.2015.7367195.

[65] F. Dai, J. Zhang, Y. Ma, and Y. Zhang, "Lenselet image compression scheme based on subaperture images streaming," in *IEEE International Conference on Image Processing*, 2015, pp. 4733–4737, doi: 10.1109/ICIP.2015.7351705.

[66] C. Jia *et al.*, "Optimized inter-view prediction based light field image compression with adaptive reconstruction," in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 4572–4576, doi: 10.1109/ICIP.2017.8297148.

[67] D. Liu, L. Wang, L. Li, Z. Xiong, F. Wu, and W. Zeng, "Pseudo-sequence-based light field image compression," in *IEEE International Conference on Multimedia Expo Workshops*, 2016, pp. 1–4, doi: 10.1109/ICMEW.2016.7574674.

[68] L. Li, Z. Li, B. Li, D. Liu, and H. Li, "Pseudo Sequence Based 2-D Hierarchical Coding Structure for Light-Field Image Compression," in *2017 Data Compression Conference (DCC)*, 2017, pp. 131–140, doi: 10.1109/DCC.2017.10.

[69] M. Magnor and B. Girod, "Data compression for light-field rendering," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 3, pp. 338–343, Apr. 2000, doi: 10.1109/76.836278.

[70] G. Wang, W. Xiang, M. Pickering, and C. W. Chen, "Light Field Multi-View Video Coding With Two-Directional Parallel Inter-View Prediction," *IEEE Trans. Image Process.*, vol. 25, no. 11, pp. 5104–5117, Nov. 2016, doi: 10.1109/TIP.2016.2603602.

[71] S. Shi, P. Gioia, and G. Madec, "Efficient compression method for integral images using multi-view video coding," in *IEEE International Conference on Image Processing*, 2011, pp. 137–140, doi: 10.1109/ICIP.2011.6115695.

[72] A. Dricot, J. Jung, M. Cagnazzo, B. Pesquet, and F. Dufaux, "Integral images compression scheme based on view extraction," in *2015 23rd European Signal Processing Conference (EUSIPCO)*, 2015, pp. 101–105, doi: 10.1109/EUSIPCO.2015.7362353.

[73] W. Ahmad, R. Olsson, and M. Sjöström, "Interpreting plenoptic images as multi-view sequences for improved compression," in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 4557–4561, doi: 10.1109/ICIP.2017.8297145.

[74] W. Ahmad, R. Olsson, and M. Sjostrom, "Towards a Generic Compression Solution for Densely and Sparsely Sampled Light Field Data," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 654–658, doi: 10.1109/ICIP.2018.8451051.

[75] J. Gu, B. Guo, and J. Wen, "High Efficiency Light Field Compression via Virtual Reference and Hierarchical MV-HEVC," in *2019 IEEE International Conference on Multimedia and Expo (ICME)*, 2019, pp. 344–349, doi: 10.1109/ICME.2019.00067.

[76] S. Zhao and Z. Chen, “Light field image coding via linear approximation prior,” in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 4562–4566, doi: 10.1109/ICIP.2017.8297146.

[77] X. Jiang, M. L. Pendu, and C. Guillemot, “Light field compression using depth image based view synthesis,” in *2017 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*, 2017, pp. 19–24, doi: 10.1109/ICMEW.2017.8026313.

[78] J. Chen, J. Hou, and L. P. Chau, “Light Field Compression With Disparity-Guided Sparse Coding Based on Structural Key Views,” *IEEE Trans. Image Process.*, vol. 27, no. 1, pp. 314–324, Jan. 2018, doi: 10.1109/TIP.2017.2750413.

[79] X. Huang, P. An, L. Shen, and R. Ma, “Efficient Light Field Images Compression Method Based on Depth Estimation and Optimization,” *IEEE Access*, vol. 6, pp. 48984–48993, 2018, doi: 10.1109/ACCESS.2018.2867862.

[80] J. Hou, J. Chen, and L. Chau, “Light Field Image Compression Based on Bi-Level View Compensation with Rate-Distortion Optimization,” *IEEE Trans. Circuits Syst. Video Technol.*, pp. 1–1, 2018, doi: 10.1109/TCSVT.2018.2802943.

[81] C. Jia, X. Zhang, S. Wang, S. Wang, and S. Ma, “Light Field Image Compression Using Generative Adversarial Network-Based View Synthesis,” *IEEE J. Emerg. Sel. Top. Circuits Syst.*, vol. 9, no. 1, pp. 177–189, Mar. 2019, doi: 10.1109/JETCAS.2018.2886642.

[82] M. Rizkallah, X. Su, T. Maugey, and C. Guillemot, “Graph-based Transforms for Predictive Light Field Compression based on Super-Pixels,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 1718–1722, doi: 10.1109/ICASSP.2018.8462288.

[83] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, “SLIC Superpixels Compared to State-of-the-Art Superpixel Methods,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, Nov. 2012, doi: 10.1109/TPAMI.2012.120.

[84] I. Viola, H. P. Maretic, P. Frossard, and T. Ebrahimi, “A graph learning approach for light field image compression,” in *Applications of Digital Image Processing XLI*, 2018, vol. 10752, pp. 126 – 137, doi: 10.1117/12.2322827.

[85] A. Aggoun, "Compression of 3D Integral Images Using 3D Wavelet Transform," *J. Disp. Technol.*, vol. 7, no. 11, pp. 586–592, Nov. 2011, doi: 10.1109/JDT.2011.2159359.

[86] X. Jiang, M. L. Pendu, R. A. Farrugia, S. S. Hemami, and C. Guillemot, "Homography-based low rank approximation of light fields for compression," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 1313–1317, doi: 10.1109/ICASSP.2017.7952369.

[87] X. Jiang, M. L. Pendu, R. A. Farrugia, and C. Guillemot, "Light Field Compression With Homography-Based Low-Rank Approximation," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 7, pp. 1132–1145, Oct. 2017, doi: 10.1109/JSTSP.2017.2747078.

[88] E. Dib, M. L. Pendu, X. Jiang, and C. Guillemot, "Super-Ray Based Low Rank Approximation for Light Field Compression," in *2019 Data Compression Conference (DCC)*, 2019, pp. 369–378, doi: 10.1109/DCC.2019.00045.

[89] I. Viola, M. Řeřábek, and T. Ebrahimi, "Comparison and Evaluation of Light Field Image Coding Approaches," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 7, pp. 1092–1106, Oct. 2017, doi: 10.1109/JSTSP.2017.2740167.

[90] D. Flynn *et al.*, "Overview of the Range Extensions for the HEVC Standard: Tools, Profiles, and Performance," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 4–19, Jan. 2016, doi: 10.1109/TCSVT.2015.2478707.

[91] *ICME 2016 Grand Challenge: Light-Field Image Compression*. http://mmspg.epfl.ch/files/content/sites/mmspl/files/shared/LF-GC/CFP.pdf.

[92] S. Kumar and R. C. Jain, "Low complexity fractal-based image compression technique," *IEEE Trans. Consum. Electron.*, vol. 43, no. 4, pp. 987–993, Nov. 1997, doi: 10.1109/30.642363.

[93] H. Huang, J. W. Woods, Y. Zhao, and H. Bai, "Control-Point Representation and Differential Coding Affine-Motion Compensation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 10, pp. 1651–1660, Oct. 2013, doi: 10.1109/TCSVT.2013.2254977.

[94] T. Wiegand, E. Steinbach, and B. Girod, "Affine multipicture motion-compensated prediction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 2, pp. 197–209, Feb. 2005, doi: 10.1109/TCSVT.2004.841690.

[95] M. Tok, M. Esche, A. Glantz, A. Krutz, and T. Sikora, "A Parametric Merge Candidate for High Efficiency Video Coding," in *Data Compression Conference*, 2013, pp. 33–42, doi: 10.1109/DCC.2013.11.

[96] A. Krutz, A. Glantz, M. Tok, M. Esche, and T. Sikora, "Adaptive Global Motion Temporal Filtering for High Efficiency Video Coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1802–1812, Dec. 2012, doi: 10.1109/TCSVT.2012.2223012.

[97] M. Tok, A. Glantz, A. Krutz, and T. Sikora, "Feature-based global motion estimation using the Helmholtz principle," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2011, pp. 1561–1564, doi: 10.1109/ICASSP.2011.5946793.

[98] R. C. Kordasiewicz, M. D. Gallant, and S. Shirani, "Affine Motion Prediction Based on Translational Motion Vectors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 10, pp. 1388–1394, Oct. 2007, doi: 10.1109/TCSVT.2007.903777.

[99] X. San, H. Cai, J. G. Lou, and J. Li, "Multiview Image Coding Based on Geometric Prediction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 11, pp. 1536–1548, Nov. 2007, doi: 10.1109/TCSVT.2007.905382.

[100] K. Müller *et al.*, "3D High-Efficiency Video Coding for Multi-View Video and Depth Data," *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3366–3378, Sep. 2013, doi: 10.1109/TIP.2013.2264820.

[101] R. J. S. Monteiro, N. M. M. Rodrigues, and S. M. M. Faria, "Disparity compensation using geometric transforms," in *3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video*, 2014, pp. 1–4, doi: 10.1109/3DTV.2014.6874751.

[102] R. J. S. Monteiro, N. M. M. Rodrigues, and S. M. M. Faria, "Geometric transforms and reference picture list optimization for efficient disparity compensation," in *International Conference on Image Processing Theory, Tools and Applications*, 2015, pp. 370–375, doi: 10.1109/IPTA.2015.7367168.

[103] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Second. Cambridge University Press, ISBN: 0521540518, 2004.

[104] P. S. Heckbert, "Fundamentals of Texture Mapping and Image Warping," EECS Department, University of California, Berkeley, UCB/CSD-89-516, Jun. 1989.

[105] J. Jain and A. Jain, "Displacement Measurement and Its Application in Interframe Image Coding," *IEEE Trans. Commun.*, vol. 29, no. 12, pp. 1799–1808, Dec. 1981, doi: 10.1109/TCOM.1981.1094950.

[106] C.-H. Cheung and L.-M. Po, "A novel cross-diamond search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 12, pp. 1168–1177, Dec. 2002, doi: 10.1109/TCSVT.2002.806815.

[107] J. R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the Coding Efficiency of Video Coding Standards - Including High Efficiency Video Coding (HEVC)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1669–1684, Dec. 2012, doi: 10.1109/TCSVT.2012.2221192.

[108] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "HEVC Complexity and Implementation Analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1685–1696, Dec. 2012, doi: 10.1109/TCSVT.2012.2221255.

[109] G. Bjontegaard, "Calculation of average PSNR differences between RD curves," *VCEG-M33*, Apr. 2001.

[110] M. J. Weinberger, G. Seroussi, and G. Sapiro, "The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS," *IEEE Trans. Image Process.*, vol. 9, no. 8, pp. 1309–1324, Aug. 2000, doi: 10.1109/83.855427.

[111] X. Wu and N. Memon, "CALIC-a context based adaptive lossless image codec," in *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, 1996, vol. 4, pp. 1890–1893 vol. 4, doi: 10.1109/ICASSP.1996.544819.

[112] H. Tang and S. I. Kamata, "A gradient based predictive coding for lossless image compression," *IEICE Trans. Inf. Syst.*, vol. E89-D, no. 7, pp. 2250–2256, 2006, doi: 10.1093/ietisy/e89-d.7.2250.

[113] X. Li and M. T. Orchard, "Edge-directed prediction for lossless compression of natural images," *IEEE Trans. Image Process.*, vol. 10, no. 6, pp. 813–817, Jun. 2001, doi: 10.1109/83.923277.

[114] M. Papadonikolakis, A. Kakarountas, and C. Goutis, "Efficient high-performance implementation of JPEG-LS encoder," *J Real-Time Image Process.*, vol. 3, pp. 303–310, 2008, doi: 10.1007/s11554-008-0088-7.

[115] Y. Li, M. Sjöström, and R. Olsson, "Coding of plenoptic images by using a sparse set and disparities," in *IEEE International Conference on Multimedia and Expo*, 2015, pp. 1–6, doi: 10.1109/ICME.2015.7177510.

[116] Y. Li, M. Sjöström, R. Olsson, and U. Jennehag, "Scalable Coding of Plenoptic Images by Using a Sparse Set and Disparities," *IEEE Trans. Image Process.*, vol. 25, no. 1, pp. 80–91, Jan. 2016, doi: 10.1109/TIP.2015.2498406.

[117] C. Conti, P. Nunes, and L. D. Soares, "Inter-Layer Prediction Scheme for Scalable 3-D Holoscopic Video Coding," *IEEE Signal Process. Lett.*, vol. 20, no. 8, pp. 819–822, Aug. 2013, doi: 10.1109/LSP.2013.2267234.

[118] C. Conti, L. D. Soares, and P. Nunes, "Light Field Coding with Field of View Scalability and Exemplar-Based Inter-Layer Prediction," *IEEE Trans. Multimed.*, pp. 1–1, 2018, doi: 10.1109/TMM.2018.2825882.

[119] A. Aaron, P. Ramanathan, and B. Girod, "Wyner-Ziv coding of light fields for random access," in *IEEE 6th Workshop on Multimedia Signal Processing, 2004.*, 2004, pp. 323–326, doi: 10.1109/MMSP.2004.1436558.

[120] P. Ramanathan and B. Girod, "Random access for compressed light fields using multiple representations," in *IEEE 6th Workshop on Multimedia Signal Processing, 2004.*, 2004, pp. 383–386, doi: 10.1109/MMSP.2004.1436573.

[121] H. Amirpour, A. Pinheiro, M. Pereira, F. Lopes, and M. Ghanbari, "Light Field Image Compression with Random Access," in *2019 Data Compression Conference (DCC)*, 2019, pp. 553–553, doi: 10.1109/DCC.2019.00065.

[122] N. Mehajabin, S. R. Luo, H. W. Yu, J. Khoury, J. Kaur, and M. T. Pourazad, "An Efficient Random Access Light Field Video Compression Utilizing Diagonal Inter-View Prediction," in *2019 IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 3567–3570, doi: 10.1109/ICIP.2019.8803668.

[123] E. Upenik, I. Viola, and T. Ebrahimi, "A Rendering Solution to Display Light Field in Virtual Reality," in *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 246–250, doi: 10.23919/EUSIPCO.2018.8553424.

[124] N. Li, J. Ye, Y. Ji, H. Ling, and J. Yu, "Saliency Detection on Light Field," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 8, pp. 1605–1616, Aug. 2017, doi: 10.1109/TPAMI.2016.2610425.

[125] A. Aggoun *et al.*, “Immersive 3D Holoscopic Video System,” *IEEE Multimed.*, vol. 20, no. 1, pp. 28–37, Jan. 2013, doi: 10.1109/MMUL.2012.42.

[126] G. Todor, *Gallery and Lightfield Data*. http://www.tgeorgiev.net/Gallery/.