Pruning and compression of multi-view content for immersive video coding

Marta Milovanovic

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2023IPPAT023

Thèse de doctorat

Pruning and Compression of Multi-view Content for Immersive Video Coding

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Télécom Paris

École doctorale n°626 École doctorale de l'Institut Polytechnique de Paris (ED IP Paris)
Spécialité de doctorat : Signal, Images, Automatique et robotique

Thèse présentée et soutenue à Palaiseau, le 6 juillet 2023, par

MARTA MILOVANOVIĆ

Composition du Jury :

Mohamed-Chaker Larabi Professeur, Université de Poitiers	Président / Examinateur
Aline Roumy Directrice de recherche, Inria	Rapporteur
Gauthier Lafruit Professeur, Université Libre de Bruxelles	Rapporteur
Frédéric Dufaux Directeur de recherche, CNRS, CentraleSupélec	Examinateur
Marco Cagnazzo Professeur, Télécom Paris et Università degli studi di Padova	Directeur de thèse
Félix Henry Ingénieur de recherche, Orange Labs et IRT b-com	Co-encadrant de thèse
Joël Jung Chercheur principal, Tencent Media Lab	Invité
Enzo Tartaglione Maître de conférences, Télécom Paris	Invité

Acknowledgements

I would like to express my heartfelt gratitude to the numerous individuals who have made the completion of this thesis possible.

I would first like to thank my academic supervisor, Marco Cagnazzo, for his unwavering optimism, support, and leadership during these three and a half years. I would like to express my gratitude to my second supervisor, Félix Henry, for making sure that everything runs smoothly for me in Orange, and for helping me grow as a researcher. I also want to express my gratitude to Joël Jung for choosing me for this PhD position and for his ongoing advice and guidance throughout these years, as it has been invaluable to me. I am incredibly grateful to Enzo Tartaglione for his boundless energy and expertise, which propelled my work to new heights and enabled me to achieve beyond what I thought possible.

I would also like to thank my thesis committee members: Aline Roumy, Gauthier Lafruit, Mohamed-Chaker Larabi, and Frédéric Dufaux, for agreeing to evaluate my work and providing me with valuable research perspectives.

In addition, I would like to express my appreciation to all my colleagues and fellow PhD students in Télécom Paris and Orange, for their mutual support and fruitful discussions during these challenging years.

Moreover, I extend my gratitude to friends that became a second family to me, both the ones who supported me before the start of my PhD journey in France, and the ones who I met along the way.

Last, but not least, I would like to thank my family and my loved ones. You taught me how to be curious, responsible, and persistent.

Résumé en français

Cette thèse aborde le problème de la compression efficace de contenus vidéo immersifs, représentés avec le format Multiview Video plus Depth (MVD). Dans le format MVD, les informations géométriques de la scène sont fournies par les cartes de profondeur associées à chaque caméra. Par rapport à la vidéo 2D classique, la vidéo immersive a besoin de beaucoup de données pour que le spectateur ait une perception adéquate de la profondeur de la scène. En raison de la demande croissante de consommation de vidéos immersives, la compression et la transmission efficaces des médias immersifs sont devenues une tâche importante pour les organismes de normalisation. Le standard du Moving Picture Experts Group (MPEG) concentré sur la transmission des données MVD est le standard ISO/IEC-23090 partie 12, appelé MPEG Immersive Video (MIV). Le standard MIV utilise des codecs vidéo 2D pour la compression des informations de texture et de profondeur de la source, qui sont prétraitées avant la compression. La nouvelle vue souhaitée est rendue à partir des informations disponibles côté décodeur en utilisant des techniques de rendu basées sur la profondeur et l'image (DIBR). Par rapport au codage vidéo traditionnel, le codage vidéo immersif est plus exigeant en termes de complexité. Outre le compromis entre le débit et la qualité, il est également limité par le débit de pixels. Le taux de pixels correspond au nombre de pixels luma par seconde qui peuvent être décodés par un décodeur hardware, afin de commencer le rendu d'une vue cible donnée. C'est pourquoi le MIV utilise le mécanisme du «pruning», qui réduit le débit de pixels et les corrélations entre les vues, en créant les morceaux d'images qui seront conservés. Ensuite, la mosaïque de morceaux d'images (patches) est emballée et transmise. Le processus de «pruning» et d'empaquetage réduit considérablement le débit et le nombre de pixels. Néanmoins, l'efficacité du «pruning» est fortement liée à la qualité de la carte de profondeur, puisque ce processus utilise des re-projections de vues avec DIBR pour découvrir les redondances. En outre, une autre approche est apparue pour le codage des vidéos immersives, appelée estimation de la profondeur côté décodeur (DSDE), qui a amélioré le système vidéo immersif en évitant la transmission des cartes de profondeur et en déplaçant le processus d'estimation de la profondeur du côté décodeur. L'approche DSDE a seulement été étudiée dans le cas de nombreuses vues entièrement transmises (sans «pruning»).

Notre travail va au-delà de l'approche DSDE et propose d'incorporer le paradigme DSDE sur le contenu qui a été traité avec le «pruning» du MIV. La méthode initiale proposée consiste à exclure un sous-ensemble de cartes de profondeur de la transmission. Ce faisant, l'étude examine l'effet distinct de la restauration de la profondeur au niveau du patch du côté du décodeur. Par ailleurs, la deuxième approche proposée présente une amélioration supplémentaire. En examinant la qualité des patchs de profondeur estimés du côté de l'encodeur, nous démontrons notre capacité à distinguer les patchs de profondeur qui doivent être transmis de ceux qui peuvent être récupérés du côté du décodeur. Nous réduisons le débit de pixels et améliorons la qualité visuelle, comme le montrent nos expériences. En outre, nous explorons l'utilisation de techniques neuronales de synthèse basées sur l'image (IBR) pour améliorer la qualité de la synthèse de nouvelles vues. Ces nouvelles techniques

donnent de bons résultats sur les objets non lambertiens et les scènes complexes, en éliminant également la nécessité de capturer et d'estimer la profondeur. Cependant, l'efficacité des méthodes IBR neuronales dépend de la disponibilité de nombreuses vues sources d'une scène, ce qui pose un défi important pour le déploiement dans les standards existants tells que MIV. Dans ce contexte, nous abordons la question du «pruning» des pixels de source pour les méthodes IBR neuronaux, en obtenant un bon compromis entre le taux de pixels et la qualité de la synthèse.

En résumé, cette thèse a montré quelques avancées possibles dans le codage vidéo immersif, en mettant l'accent sur le «pruning» du contenu de source. Nous avons amélioré la conception globale du système de codage vidéo immersif en proposant l'approche DSDE au niveau du patch, ce qui a permis d'obtenir un gain de 4.63% BD-rate pour le Y-PSNR en moyenne. En outre, nous avons démontré pour la première fois que la synthèse neuronale fournit elle-même les informations nécessaires au pruning du contenu, ce qui a permis d'améliorer la qualité de la synthèse de vues de 3.6dB en moyenne. Notre travail encourage donc une adoption plus large du standard MIV et un développement plus poussé des IBR neuronaux dans un tel contexte.

Contents

Chapter 1

Introduction

1.1 Context

Recently, the popularity of commodities such as virtual, augmented, and extended reality has seen a significant rise, followed by an increased interest in the evolution of the Internet toward metaverse [1]. The technologies that aim to offer a virtual presence to the user are called immersive imaging technologies [2, 3, 4]. They aim at facilitating free navigation of a user in a 3D scene which can be real and enhanced with virtual people and objects, or fully virtual [5]. While being particularly useful for many fields, such as education, healthcare, marketing, sales, *etc.* [6], these emerging use cases induce new challenges in the area of data compression for enabling data transmission over current networks.

One of the traditionally most popular formats for enabling the immersive experience of a user in a limited volume, with six degrees of freedom (6DoF), is multi-view video plus depth (MVD) [7, 8, 9]. In this context, a 3D scene can present natural content (NC), captured with multiple cameras with an arbitrary arrangement, or computer-generated content (CG), created with imaging software. Each captured or created viewpoint has its own *texture* video and corresponding *depth map*. A texture video represents a conventional color video, while in its depth map, each pixel (associated with its color counterpart) represents the distance between the captured object and the camera. Moreover, the term *view* represents a certain camera viewpoint. The target viewport is rendered from available information at the decoder side utilizing depth image-based rendering (DIBR) techniques [10, 11, 12]. The dense sampling of the scene, obtained with a large number of cameras and high resolutions and needed to ensure a satisfactory immersive user experience [13], comprises a vast amount of data whose transmission over the networks poses a difficult challenge. An example for an MVD case: 10 streams of high-definition texture video with dimension 1920×1080, and 50 frames per second (fps) would require, without compression, a total bitrate of almost 21 Gbps [14]. Therefore, an efficient algorithm is essential to eliminate the redundancies among the videos of the captured scene.

Immersive video coding techniques date back to MPEG-2 extension and its multi-view profile (MVP) [15], followed by its AVC successor called multi-view video coding (MVC) profile [16]. Some of the methods for multi-view transmission rely on simulcast coding, where all videos are coded independently with some of the legacy video codecs such as AVC [17] or HEVC [18]. However, simulcast coding is not optimal for produced bitrate and it requires an enormous amount of decoding devices on the client side (at least dozens), which is unfeasible considering typical consumer devices. This is further discussed in Chapter 2.

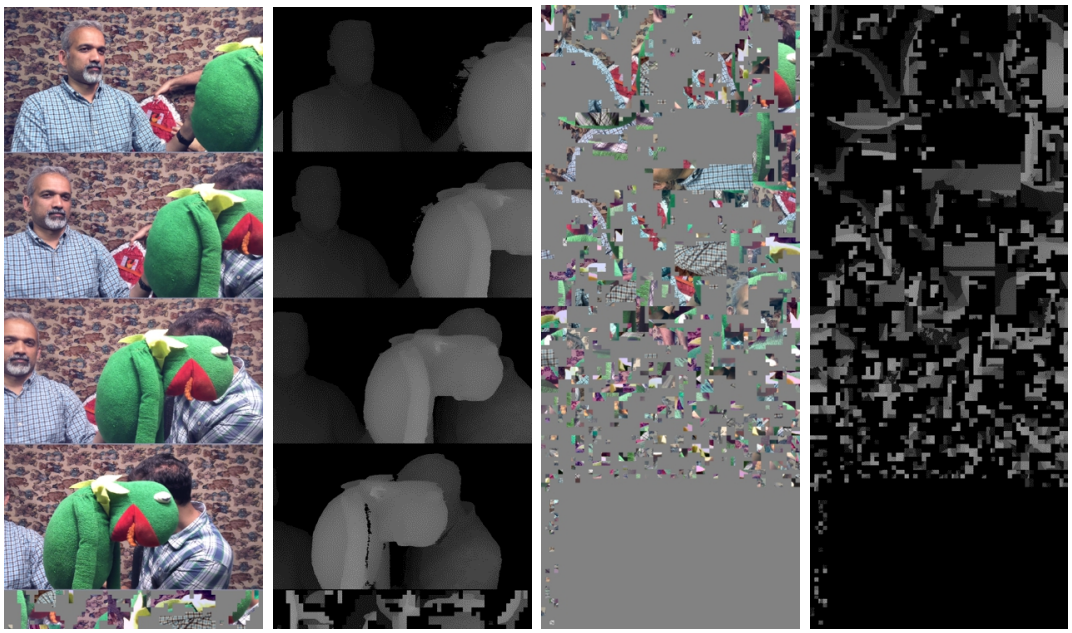
These approaches, as well as the recent ones called MV-HEVC and 3D-HEVC [19], aim at building on an already existing 2D video codec (in this case HEVC) and enable inter-view prediction among different views, which helps to reduce the existing

redundancies among views. Still, they suffer from parallelization issues related to the strong dependencies between the coded views and need an excessively large number of decoding devices at the client end. Another important constraint for an immersive video codec is the *pixel rate*, which reflects the number of luma pixels per second that can be decoded by a hardware decoder, in order to commence the rendering of a given target view. Previously mentioned standards fail to adhere to pixel rate constraints (for a use case of high-end mobile devices, 32 Megapixels at 30 fps) as they require transmission and decoding of dozens of views [20]. Furthermore, these standards had limitations in terms of camera arrangements, thus showing good performance only for linearly arranged cameras [14].

The Moving Picture Experts Group (MPEG) is currently addressing the challenges of immersive video coding in the scope of the MPEG-I project [21]. One of the standards from this project is ISO/IEC-23090 Part 12 called MPEG Immersive Video (MIV) [22], which is focused on the transmission of MVD data. The first edition of the MIV standard was released in July 2021 [23]. This standard does the preprocessing of source texture and depth videos, as detailed later on. Subsequently, these videos are compressed using any 2D video codec. The received video streams are decompressed at the client side, and the rendered virtual view is obtained using a suitable DIBR technique. MIV is detailed in Chapter 3, but we give a quick overview of its main concepts in the following. MIV provides a framework that adheres to the pixel rate constraints, and simplifies the inter-view redundancy removal, as compared to previously discussed approaches. The reference software for MIV is the Test Model for MPEG Immersive Video (TMIV) [24]. TMIV processes the source data and generates the *atlases*, which are collections of *patches*, where patches are 2D rectangles originating from different views.[1] Thus, TMIV involves many challenging tasks: selecting the most important views among source views (view labeling), removing the redundancies between the views that were not selected (pruning), constructing the atlases that will be sent (packing), and rendering the target viewport. View labeling classifies the views as basic, which are transmitted completely, or additional, which are pruned and transmitted partially. The pruning process decides which pixels of the views to send, *e.g.*, the parts that are missing in a particular additional view because they are occluded in basic and other additional views. However, the rectangular patch region is multiplied with the pruning binary mask, which refines the patch shape and saves only the necessary pixels. The process of pruning and packing significantly reduces the bitrate and pixel rate. Nevertheless, the effectiveness of the pruning is closely tied to the quality of the depth map, since the pruning process employs view re-projections with DIBR to discover the redundancies. A simple example of texture and depth map atlases is given in Fig. 1.1. In Fig. 1.1a we can see four packed basic views, together with small patches, at the bottom of the texture atlas 1, while Fig. 1.1c shows the atlas which consists solely of the patches. Analogously, we can observe their corresponding depth atlases.

In addition, a novel technique called decoder-side depth estimation (DSDE) has emerged [25], which eliminates the need to transmit depth maps and shifts the depth estimation process to the decoder side. This technique is adopted into MIV standard as Geometry Absent (GA) profile [26]. DSDE is beneficial because it avoids harmful coding of depth data with legacy 2D video codecs, which do not possess adequate depth coding tools.

[1] In this thesis, from now on we use the term patches to denote the pruned, partial image regions. It is not to be confused with the same term in the MIV standard, which denotes both partial regions and full views, packed into atlases.



(A) Texture atlas 1. (B) Depth map atlas 1. (C) Texture atlas 2. (D) Depth map atlas 2.

FIGURE 1.1: An example of texture and depth map atlases.

1.2 Motivation and contributions

Due to the complexity and limited adoption of previous video-based solutions (like MV-HEVC and 3D-HEVC), the MIV standard has adopted a simpler and easily deployable paradigm for immersive video coding, as described earlier. However, even after a few MPEG meeting cycles with software improvements, the synthesis results obtained with TMIV were not completely satisfactory. Many artifacts are visible, especially when it comes to complex content, as shown in Fig. 1.2.

The work of this thesis started at the beginning of the MIV standardization, with the goal to increase the view synthesis quality and compression efficiency. Therefore, we focus on the following issues. Firstly, the coding of the depth maps with 2D codecs is damaging and inefficient. Secondly, pruning, which reduces the pixel rate and inter-view correlations, is highly dependent on the depth map quality. Our goal is to improve the compression and pruning of multi-view content for immersive video coding.

MIV is explored in MPEG in multiple possible configurations: transmitting a few full views (textures and depth maps) and numerous smaller patches, transmitting just some amount of full views (textures and depth maps), and transmitting double the amount of full views (just textures) while retrieving the depth maps at the decoder side. However, the question is, which configuration is the most efficient, and can we propose new configurations for additional benefits? Thus, taking inspiration from the DSDE concept, we explore the case of the main MIV framework, which includes pruning. DSDE is created and tested in a particular scenario, where several full texture views are coded with MIV, and their depths are recovered at the receiver side. However, this thesis goes beyond that and inspects multiple improvements of such a system. Notably, we are interested in reducing the pixel rate even further by introducing the depth estimation done on the content that has been pruned with MIV and does not consist anymore of the rich 3D scene.

Furthermore, we consider the neural image-based rendering (IBR) techniques to

FIGURE 1.2: Visual example of the synthesis result with TMIV for the Carpark video sequence.

improve the synthesis quality, since they render non-Lambertian objects more successfully than DIBR methods, and eliminate the burden of capturing or estimating depth. However, neural IBR methods require many available source views of a given scene to achieve good synthesis quality, creating a big obstacle to their deployment using existing standards (such as MIV). Thus, in this context, we tackle the input pixel pruning issue for neural IBRs.

To summarize, the work conducted during this thesis aims at further development on top of the MIV standard, enabling its broader adoption. It gave us an opportunity to investigate many possible improvements of an immersive video coding system, including the following:

- Atlas compression, where we explore the impact of different encoders and coding tools on the immersive video coding efficiency [27] (Chapter 4),
- Depth estimation techniques, where we observe the influence of different depth estimators on the immersive coding efficiency [26] (Chapter 5) and synthesis quality without compression [28] (Appendix A),
- Source content pruning, which is our main contribution. Here, pruning of the input represents a new way of compressing the inter-view redundancies. We explore avoiding the transmission of some depth patches in TMIV and retrieving them at the decoder side [29, 30] (Chapter 6). Moreover, we also investigate the impact of input texture pruning on the synthesis quality obtained with a neural image-based renderer [31] (Chapter 7).

1.3 Structure of the thesis

The manuscript is structured as follows.

Chapter 2 introduces the general principles of video coding, immersive video representations, and multi-view standards. The classical and learning-based depth

estimation methods are presented. Depth image-based rendering and image-based neural rendering techniques are explained.

Chapter 3 details the general principles of the MIV standard and its reference software. Decoder-side depth estimation (DSDE) concept is described. Test Conditions for MIV are listed (datasets, software, metrics), and the performance of the main MIV anchor with respect to other MIV anchors is demonstrated.

Chapter 4 presents results [27] which highlight that MIV is compatible with any state-of-the-art 2D codec. Furthermore, we show that an improvement of the coding performance can be achieved by utilizing screen content coding tools, especially for depth atlases.

Chapter 5 gives a study [26] on depth estimation tools for the MIV DSDE system. The analysis of synthesis results obtained with different classical and learned depth estimation techniques shows that various depth estimation methods may be suitable for the MIV DSDE framework.

Chapter 6 presents two studies on reducing the transmission of patch depth data in the context of MIV. We investigate the possibility to avoid the transmission of some patch depths that originate from the additional views. The first study [29] presents pDSDE, a method that focuses on the isolated impact of the patch-level depth recovery at the decoder side, while not changing the pruning process. The second study [30] goes a step further and proposes a method called hpDSDE, that ensures a reliable patch depth selection and recovery of depths at the decoder side.

Chapter 7 introduces a method [31] for input pixel pruning, called LeHoPP, that learns how to prune the pixels and can be utilized with neural image-based synthesizers in a multi-view setup without retraining the model. In this setup, we examine the importance of each input pixel concerning the rendered view, and we avoid the use of irrelevant pixels.

Chapter 8 draws conclusions on the thesis and gives a discussion on the potential future work and research perspectives.

Appendix A gives a brief overview of a study [28] which investigates the performance of different conventional and learned depth estimators in terms of MIV synthesis quality.

Appendix B gives a full list of publications issued during the work on this thesis.

Chapter 2

Background and State of the Art

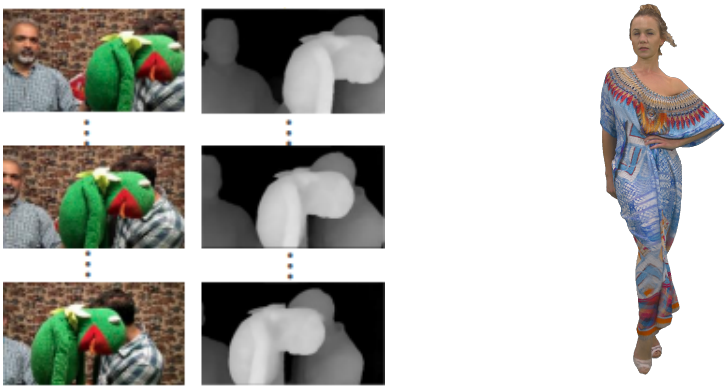
In this chapter, we provide an overview of various technologies used in immersive video coding. We first discuss the different formats used for representing volumetric videos and then describe the block-based hybrid video coding structure for compressing 2D videos. Next, we delve into the different immersive video coding schemes available. We then proceed to explain the various techniques used for depth estimation, which include both classical and learning-based approaches. Finally, we provide an overview of virtual view synthesis techniques, with a focus on depth image-based rendering and neural image-based rendering approaches.

2.1 Introduction

The realm of video services is undergoing a transformation, shifting away from conventional two-dimensional videos toward virtual and augmented reality applications. These technologies enable viewers to perceive a scene as if they are inside the video. However, they still pose a challenge to the industry: to provide a novel viewpoint to the viewer, a high amount of data needs to be processed and transmitted over the networks, which poses constraints that include power consumption, transmission bandwidth, and hardware capabilities.

Over the course of immersive video development, many formats for the representation of immersive images and videos emerged. The oldest and simplest representation for 3D video **is stereoscopic 3D video** [3]. It is based on the stereopsis principle and contains two videos capturing the same scene. Stereopsis is based on a binocular disparity, which refers to the distance in image location of an object seen by the left and right eyes, caused by the displacement between the eyes (parallax) [32]. The brain uses binocular disparity to extract depth information. Typically, the optical axes of the cameras are parallel and the cameras also share the image plane. A disparity between the cameras is the pixel displacement in image location of an object seen by the left and right camera, and it creates the impression of depth. A simple extension of stereoscopic 3D video is **multi-view video**, where a system of cameras obtains multiple views of the same scene simultaneously.

Another representation, which is very common for immersive video, is **multi-view plus depth** (MVD), which is the format where each view (also called texture view) is accompanied by a corresponding depth map. The depth map is a single-channel image where each pixel contains the distance of the object to the camera. Thus, depth maps deliver information about the physical scene geometry, *i.e.*, relative positions of objects and cameras. An example of the MVD content is given in Fig. 2.1a. The depth maps for computer-generated content are obtained directly from the 3D scene model. The depth maps for natural content can be obtained either with sensors or in the process of depth estimation, as detailed in Section 2.4. MVD format



(A) MVD. (B) Point cloud.

FIGURE 2.1: An example of the MVD [33] and point cloud [34] data.

enables the use of depth-image-based rendering techniques, which interpolate the virtual views and facilitate free navigation, as explained in Section 2.5.1.

Another format for 3D volumetric (immersive) data representations is **point cloud.** A point cloud is a set of points with (x, y, z) coordinates that represent the point cloud geometry, and can, in addition, contain the attributes (*i.e.,* color, reflectance, and normals) [35]. In addition, point clouds are classified as static or dynamic, depending on the absence or presence of the temporal dimension, respectively. An example of the point cloud is given in Fig. 2.1b.

Throughout this thesis, we focus on the MVD content, with the exception of Chapter 7, where we consider multi-view video with only texture data.

In this chapter, we introduce the reader to the important aspects of immersive video coding. The existence of many formats for immersive video representation resulted in different solutions for data compression. To better understand the influence of 2D video codecs on the compression of volumetric videos, we will first give an overview of the block-based hybrid video coding structure, and later we will briefly describe the screen content coding tools and recently developed video codecs outside of MPEG, tested in the scope of this thesis. We follow it with an overview of immersive video coding schemes: simulcast, multi-view, and novel volumetric video-based approaches. Afterward, Section 2.4 explains the depth estimation techniques, first giving the perspective projection equations, and then detailing the classical and learning-based approaches. Finally, Section 2.5 gives an overview of virtual view synthesis, notably the depth image-based rendering and neural image-based rendering approaches.

2.2 Hybrid 2D video coding

The pipeline for 2D video transmission is shown in Fig. 2.2. The source video sequence can be pre-processed (*e.g.,* trimming or color correction), and afterward, it is encoded (usually with lossy compression) and transformed into a video bitstream. The bitstream is a very compact representation of the source video, which is transmitted over the channel. It is decoded at the receiver, and given to the post-processor, for a potential color correction, re-sampling, *etc.* Finally, we obtain a reconstructed video sequence that needs to be in a suitable color format for a target display [36].

Modern video codecs rely on the notions of spatial and temporal prediction. **Spatial prediction** is, in essence, decorrelating the signal by exploiting the redundancies

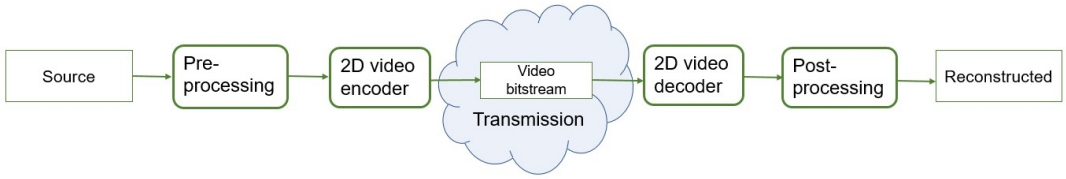


FIGURE 2.2: General block diagram of a video transmission system.

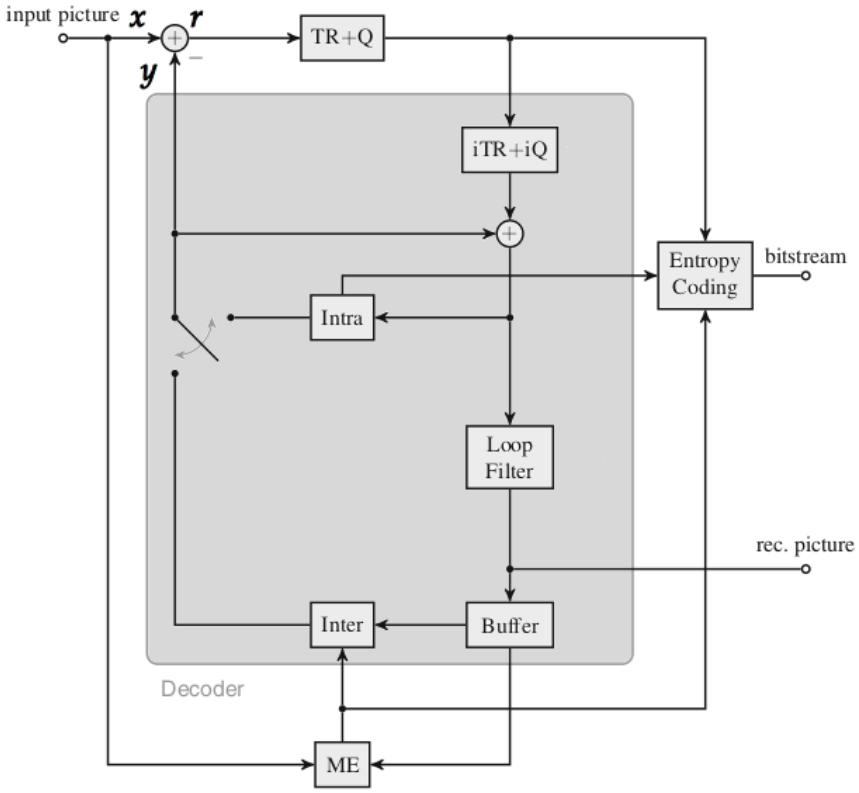


FIGURE 2.3: Diagram of a block-based hybrid codec [36].

inside of a particular frame, and representing them with geometric models. This type of prediction is called intra prediction. **Temporal prediction** is removing the statistical redundancies which appear when observing the video in the temporal domain. It models the displacement of the objects in a scene in different frames, by deploying motion estimation (ME). Motion estimation compares the original block with blocks in a reference frame (it could be some of the previous or even some of the future frames). In that case, the block is represented with a motion vector, which shows how the block from the reference should be moved to the original frame, to create the prediction. This type of prediction is called inter prediction.

A video consisting of a certain number of frames can be divided into one or multiple coded video segments that can be encoded separately. The order in which pictures are reconstructed at the decoder, and that signals which pictures can be used for reference, is determined by the coding order. The coding structure is made up of a series of pictures in the sequence that are encoded in a specific order and have defined dependencies between them, called group of pictures (GOP).

The inter and intra prediction models create the prediction signal (y) of the original one (x), which introduces an error, called a residual (r). The residual is calculated as follows:

$$r = x - y. \tag{2.1}$$

Thus, the residual is being transmitted by the encoder.

All modern video codecs (including the ones used in the scope of this thesis) have a hybrid video coding scheme in common, which is shown in Fig. 2.3. The name hybrid comes from integrating temporal prediction between the frames (pictures) of the video sequence with transform coding applied on the prediction error [36].

Transform coding is an essential module for coding the residual. The motivation for using the transform module lies in the nature of the residual signal, that is the low frequency of its energy. Moreover, transform coding facilitates the lossy coding of the residual signal via quantization of the transform coefficients, as expressed through the quantization parameter (QP). Together, they are represented as the "TR+Q" block in Fig. 2.3, while the inverse transformation and quantization, applied in the decoder for the signal reconstruction, are "iTR+iQ".

In the process of video encoding, various coding options such as different picture partitioning, prediction, and transformation are employed. These options are assessed in a rate-distortion (*R-D*) optimization process. Here, D represents a distortion of the signal obtained by coding the considered block in a particular mode or parameter, R is the transmission rate (number of bits) for representing a block in a given mode or rate needed for coding a given parameter. If P denotes the picture (partitioned into coding blocks), and M denotes the possible prediction and coding modes, one may formulate the constrained problem for the encoder to find the best coding mode set M_{opt}, which is subject to a given rate constraint R_c:

$$\min_{M}(D(P, M)), \text{ with } R(P, M) < R_c. \tag{2.2}$$

We can transform this into an unconstrained problem, to determine which one results in the lowest R-D cost J using a Lagrangian parameter λ that is calculated based on the given QP, as follows:

$$M_{opt} = \arg\min_{M}\Big(J(P, M|\lambda)\Big), \tag{2.3}$$

having the joint cost criterion equal to:

$$J(P, M|\lambda) = D(P, M) + \lambda \cdot R(P, M). \tag{2.4}$$

Finally, after coding decisions take place, the corresponding syntax elements for all the modes, *etc.* are coded into the bitstream with an entropy coding technique (such as context adaptive variable length coding, or context adaptive binary arithmetic coding).

2.2.1 Screen content coding tools

It has been noticed that some of the available coding tools are having poor performance when it comes to certain types of content. Notably, computer-generated text, graphics, and animations, as well as their combination with camera-generated content, which are referred to as screen content, are particularly challenging [37]. This is due to the different characteristics in their signal (repeated patterns, sharp edges,

absence of noise, *etc.*). As a result, an extension profile of HEVC, and VVC were developed to provide support for such content.

The VVC screen content coding tools employed in this thesis (see Chapter 4) are intra block copy (IBC), block-based differential pulse-code modulation (BDPCM), and palette mode. They are briefly summarized below.

IBC is a technique that applies block matching for intra prediction having the same frame for the current frame and the reference. This means that it is searching for the repetitions inside a particular frame. To that end, a limited region is designated around every block, and a search is conducted within that region to identify the block with the greatest similarity to the present block.

Palette mode is a technique that considers coding the content with distinct colors (highly saturated and unique colors). This method consists of palette table derivation, coding of the palette table denoting the samples, and coding of the palette index for each coding unit.

BDPCM is a method that proceeds without transform coding, and is based on the residual DPCM method from HEVC [38]. It is an intra prediction mode that aims to predict the residual signal in the pixel domain using a sample-based approach. The residual samples are coded differentially and signaled so that at the decoder side, each row or column can be reconstructed by summing weighted DPCM residuals.

2.2.2 Codecs outside of MPEG

Ever since its appearance, the MPEG organization successfully developed many efficient coding standards, including the most recent ones, such as those already mentioned - **HEVC** and **VVC**. However, the hybrid video coding scheme, as well as the methods for spatial and temporal prediction, are not only used in the MPEG family of standards but also in modern video codecs which are used all over the world.

Since we do not know which codec will dominate the market, and MIV is designed to benefit from itself being codec-agnostic, in Chapter 4 of this thesis, we also test the atlas compression performance of codecs outside of MPEG. We test the AV1 standard [39] from Alliance for Open Media [40], as well as the AVS3 standard [41] from Audio Video Coding Standard Workgroup of China's [42]. **AV1**, a video coding standard that is open and free of royalties, is commonly regarded as the primary rival to VVC and HEVC. Although its main application is video transmission over the Internet, AV1 is believed to be a versatile codec capable of coding any type of video signal [43]. **AVS3** is the third generation of Audio Video Standard in China, focused on improving the coding for higher resolution content and more efficient parallel processing architectures [41].

2.3 Immersive video coding schemes

Depending on the chosen immersive (volumetric) video format, one may choose a different coding paradigm. In this section, we will focus on the coding schemes utilized to compress MVD data.

2.3.1 Simulcast coding

The first paradigm for MVD video coding is simulcast coding. In a simulcast setup, every video is encoded independently with some of the legacy video codecs. In the case of MVD content, each texture and depth video is encoded separately. Here, the similarities among the views are not considered in the encoding process [14, 3].

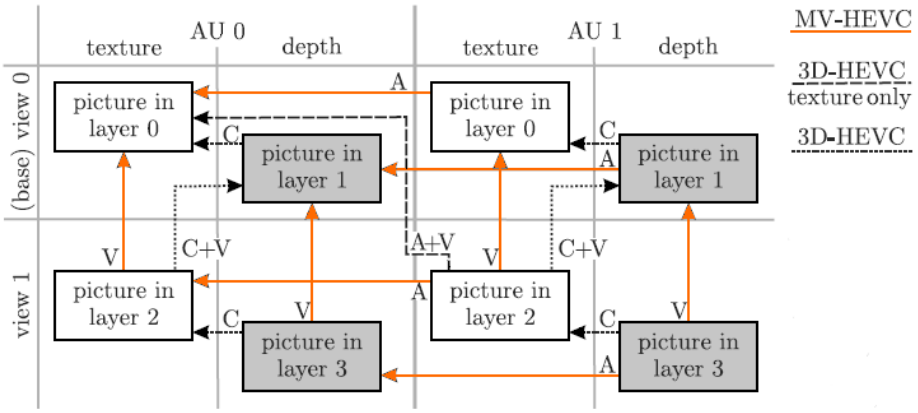


FIGURE 2.4: Different inter-picture predictions [44] ©2016 IEEE.

At the decoder side, novel views can be generated using the depth-image-based rendering techniques (see Section 2.5.1). The benefit of a simulcast approach is its simplicity since it is using off-the-shelf 2D video codecs. Moreover, simulcast does not include more complex predictions among the texture videos or between texture and depth videos, and can perform compression in a distributed fashion. However, it is not an optimal solution in rate-distortion terms, since it is not considering the similarities among multiple views of a scene.

2.3.2 Multi-view and 3D video coding

Another paradigm for immersive video coding is aiming to exploit additional redundancies that appear when the same scene is being observed from multiple perspectives. Similarly to the simulcast pipeline, novel virtual views can be generated using the DIBR techniques (see Section 2.5.1) at the decoder side. The HEVC extensions designed to support multiview and 3D video are known as Multiview High Efficiency Video Coding (MV-HEVC) and 3D High Efficiency Video Coding (3D-HEVC), respectively.

MV-HEVC and 3D-HEVC extensions employ a multilayer approach for predictive coding, achieving strong bitrate savings via the tools for inter-view prediction and inter-component prediction, respectively [44]. Here, different layers are multiplexed into one final bitstream and they can depend on each other. A view is denoted by the layers that correspond to texture and depth videos belonging to the same camera perspective. A component is denoted by the layers that carry the same type of information, (texture or depth). An access unit (AU) contains all pictures that are related to the same capturing or display time instance.

Inter-view prediction is the prediction done between an observed (texture) view and its reference, which belongs to another (texture) view. It is deployed in MV-HEVC. **Inter-component prediction** is exploiting the dependencies between the texture video and its corresponding depth. It is used in 3D-HEVC, together with the inter-view prediction.

Fig. 2.4 depicts different possible inter-picture predictions between the views: A denotes an access unit, V denotes the notion of views, and C components. MV-HEVC includes only the inter prediction (temporal prediction), and inter-view prediction in the same access unit. The base view contains only the temporal references (A, prediction from the reference AU0 to AU1), while other views have temporal, but

also reference pictures from other views (V, from the reference in view0 to view1). 3D-HEVC texture-only coding additionally enables the inter-view prediction across different AUs (A+V, from the reference AU0 in view0 to AU1 in view1). 3D-HEVC texture and depth coding enables the inter-component prediction between textures and depths, in the same AU. The first example is C, prediction from the reference texture in view0 to depth in view0. The second example is C+V, combined inter-component and inter-view prediction, from reference depth in view0 to texture in view1.

3D-HEVC also deploys depth coding tools, which deal with the distinct features of depth maps (such as large homogeneous areas separated by sharp edges). These methods include new intra-picture prediction and residual coding. Furthermore, additional depth tools have been introduced, enabling the prediction of motion and partitioning information from texture layers, or inter-view prediction of motion.

The benefit of MV-HEVC is that it can be implemented without changing the syntax or decoding process of single-layer HEVC below the slice-level header. Therefore, it allows the reuse without the big changes required to build the MV-HEVC decoder. However, it requires having all the data centralized in one place (encoder and decoder), which is difficult to deploy in practice, and it does not conform to the pixel rate constraints for immersive video [20]. Moreover, inter-component prediction enabled by 3D-HEVC brings additional complexity to the encoder, making its deployment even more challenging. Furthermore, both MV-HEVC and 3D-HEVC have limited efficiency depending on the content characteristics. Their inter-view prediction is computed via block-based disparity, which means that the coding tools are efficient only in the case of 1D linear and coplanar camera arrangement [44].

All of the previously mentioned drawbacks caused the low adoption of these MVD standards in the industry, which lead to the development of a simpler approach for coding the MVD data, which will be covered in detail in Chapter 3, and which is the main object of this thesis.

2.3.3 Visual volumetric video-based coding

Due to the disadvantages of basic simulcast and multi-view and 3D video coding for MVD data, a new approach was created, which is supposed to ease the adoption in the industry and enable the practical use case. With that goal, the constraints on pixel rate and number of decoders at the receiver are set, which follow the experimental results stating that on a current high-end mobile device, it is possible to simultaneously decode up to four ultra HD resolution streams at 30 frames per second [20]. This approach results in just a few full views and only the parts of other views (patches) that are missing for a high-quality rendering, by removing redundancies via re-projections. This solution is called ISO/IEC 23090-12 MPEG Immersive Video (MIV) standard [23]. The output of MIV is encoded using already available 2D video codecs, and rendering is done with a non-normative DIBR technique. MIV will be exhaustively presented in Chapter 3.

Recently, several tools have also been proposed in MPEG for efficient point cloud data compression. Notably, there are two distinct technologies, one being ISO/IEC 23090-5 Video-based Point Cloud Compression (V-PCC), and another being ISO/IEC 23090-9 Geometry-based Point Cloud Compression (G-PCC) [45].

V-PCC relies on the idea of projecting the point cloud onto 2D planes, which simplifies the problem of point cloud compression by reducing it to image and video coding. This approach is beneficial because it is easy to deploy. It leverages existing state-of-the-art video codecs, and additional V-PCC bitstream is easy to decode since

it solely describes the information needed for subsequent (non-normative) rendering. V-PCC is proven to perform better than G-PCC on dynamic point clouds.

On the other hand, G-PCC encodes a point cloud straight from the 3D domain. It separately codes the geometry and attributes, where attribute coding leverages the decoded geometry, to gain coding accuracy [35].

Both MIV and V-PCC aim to encode their data in a video-based fashion, therefore, they have been aligned in a common specification called Visual Volumetric Video-based Coding (V3C). V3C gives extension mechanisms for V-PCC and MIV. Thus, since their terms and definitions are aligned, in this thesis we will use the words *geometry* and *depth*, as well as *attribute* and *texture* interchangeably.

The next section will detail the important tools for obtaining depth maps in the process of depth estimation.

2.4 Depth estimation

The multi-view plus depth video format indicates having depth maps alongside texture videos. Acquiring depth maps can be done by depth cameras and depth sensors (such as structured light and time-of-flight [46]). These are the so-called active methods, and they are limited by their low resolution, limited measurement range, and high cost [47]. To overcome these challenges, depth estimation is an alternative (passive) approach to computing depth maps from the images captured by image sensors. Depth estimation involves the techniques used to extract the geometry of a 3D scene from one or more cameras. Depth estimators can be classified as monocular, using a single view to obtain its depth value by extracting depth cues [48], and (multi-view) stereo, dealing with two or more views, assuming that a particular pixel can be found in another view [49].

Depth estimation methods in the literature often only address a subset of the requirements needed for an immersive video system (real-time applications [50, 51], inter-view consistency [49], depth super-resolution [52], wide-baseline setup [53]). Other problems, such as temporal consistency for videos, domain shift between natural and computer-generated content, high source image resolutions, compliance for different camera setups and image/video projections, *etc.* are rarely addressed [26, 54].

The approaches studied in the MPEG standardization for the development of novel immersive video technologies, such as DERS [55] and IVDE [56], are meant to be versatile and adapt to a general purpose of an immersive video system. This thesis evaluates the performance of these approaches in a given immersive video framework. We also study the performance and benefits of two approaches outside of MPEG, which are learning-based, called GwcNet [57] and GA-Net [58]. Although these methods were not initially developed for view synthesis applications, they were available and recognized in the literature as high-performing. We evaluate them in the context of depth image-based rendering for immersive video, and also observe their performance, as compared to the MPEG approaches.

The rest of the section will recall the concepts of 3D geometry, needed to describe the depth estimation problem, and detail the mentioned non-learning-based (classical) and learning-based approaches employed in Chapter 5.

2.4.1 Perspective projection

The goal of multi-view stereo techniques is the following: given several images of the same scene (or object), compute the representation of its 3D shape. For multi-view stereo algorithms to function, it is necessary for each input image to have a corresponding camera model that explains the process of projecting a 3D point in the real world into a 2D pixel position in a specific image. The most frequently utilized camera model is the *pinhole or perspective camera model* [59], which we detail here.

Firstly, we will derive the perspective projection from world to camera coordinates, as illustrated in Fig. 2.5. Let us have a look at a point P_w with world coordinates (X_w, Y_w, Z_w). To convert these to camera coordinates $P_c = (X_c, Y_c, Z_c)$, we use the equation in homogeneous coordinates:

$$\begin{bmatrix} P_c \\ 1 \end{bmatrix} = \begin{bmatrix} R & T \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \begin{bmatrix} P_w \\ 1 \end{bmatrix}, \tag{2.5}$$

where the vector T of size 3×1 describes the world origin W in the camera coordinate system with the center in C, and the rotation matrix R of size 3×3 describes the orientation of the world coordinate frame with respect to the camera frame. The matrix $[RT]$ is called pose matrix, or denoted as extrinsic parameters matrix.

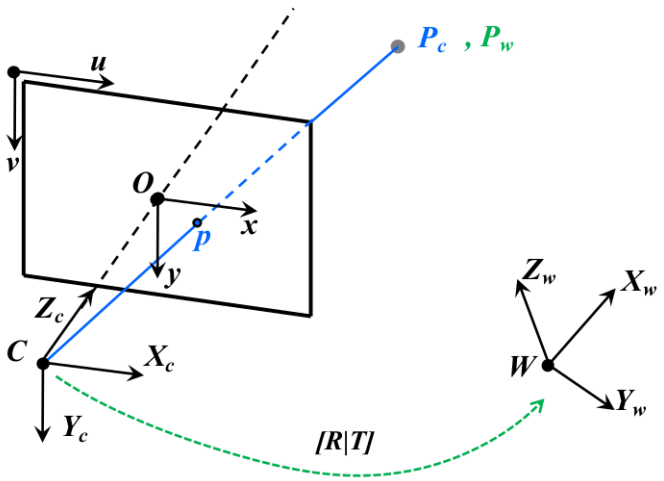


FIGURE 2.5: Projection from world to camera coordinates [60].

Secondly, we will describe the projection of the point P_c from the camera frame to pixel coordinates (u, v). To do this, we utilize the following conversion:

$$\begin{bmatrix} \lambda u \\ \lambda v \\ \lambda \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = K \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}, \tag{2.6}$$

where K represents the calibration matrix, or instrinsic parameters, f_u is the focal length in the u-direction, and f_v is the focal length in the v-direction. It is usually assumed that pixels are square shaped, *i.e.*, $f_u = f_v = f$. In Fig. 2.5, f is the distance between the camera system origin C and the so-called principal point O, which is located at the intersection of the camera optical axis and the image plane. For the linear mapping from 3D to 2D, we use homogeneous coordinates by introducing a scale factor λ, which is usually equal to 1.

Finally, combining the Eq. (2.5) and Eq. (2.6), we get the perspective projection from the world coordinates to the image plane:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} R & T \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}, \tag{2.7}$$

and we call the matrix $K[R\ T]$ the projection matrix. Having defined the projection process, we now proceed to define the concept of stereo disparity and explain how depth can be computed.

2.4.2 Stereo disparity and depth computation

Let us consider a point P in 3D space. Its depth is defined as the distance between the optical center of the camera lens and the plane which contains the point P and is perpendicular to the camera optical axis [61]. If the world coordinates of the point P are given by $P = \begin{bmatrix} X & Y & Z \end{bmatrix}^T$, a projection equation in homogeneous coordinates to the point $p = \begin{bmatrix} u & v \end{bmatrix}^T$ in the image plane is as follows:

$$z \begin{bmatrix} p \\ 1 \end{bmatrix} = K \begin{bmatrix} R & T \end{bmatrix} \begin{bmatrix} P \\ 1 \end{bmatrix}, \tag{2.8}$$

where z is the distance of the point P from the plane perpendicular to the camera optical axis that contains the camera center. Depth in z-distance format is represented in arbitrary units.

Now, let us consider a stereoscopic acquisition system, where the two cameras are located side by side and have identical intrinsic parameters, we have:

$$K_L = K_R = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.9}$$

Let us assume that both cameras look in the direction of z-axis and that the left camera defines the common 3D coordinate system (*i.e.*, $R_L = R_R = I$), while the right camera is shifted by b units along the x-axis: $T_L = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$, and $T_R = \begin{bmatrix} b & 0 & 0 \end{bmatrix}^T$. The value b is called the baseline, and it is equal to the distance between the optical centers of the two cameras. Two cameras capturing the point P in the scene are projecting it onto the image plane of the first camera and the second camera as the point $p_L = \begin{bmatrix} u_L & v_L \end{bmatrix}^T$ and the point $p_R = \begin{bmatrix} u_R & v_R \end{bmatrix}^T$, respectively. Substituting these values into the two perspective projection equations gives:

$$z_L \cdot p_L = \begin{bmatrix} f \cdot X + u_0 \cdot Z \\ f \cdot Y + v_0 \cdot Z \\ Z \end{bmatrix}, \tag{2.10}$$

$$z_R \cdot p_R = \begin{bmatrix} f \cdot (X - b) + u_0 \cdot Z \\ f \cdot Y + v_0 \cdot Z \\ Z \end{bmatrix}. \tag{2.11}$$

Combining the equations (2.10) and (2.11) we get:

$$\begin{aligned} z_L &= z_R = Z \\ u_L &= f \cdot \frac{X}{Z} + u_0 \\ u_R &= f \cdot \frac{X-b}{Z} + u_0 \\ v_L &= v_R = f \cdot \frac{Y}{Z} + v_0 \end{aligned} \tag{2.12}$$

Thus, the cameras project the point P on the same row (vertical y-axis) $v_L = v_R$, while horizontal shift (on x-axis), called *disparity*, is equal to:

$$d = u_L - u_R = f \cdot \frac{b}{Z} \tag{2.13}$$

Therefore, the disparity d refers to the difference in image location of the same 3D point when projected under perspective to two different cameras. As it can be seen from Eq. (2.13), we can fully define the 3D position of a point by knowing the disparity, camera baseline, and focal length.

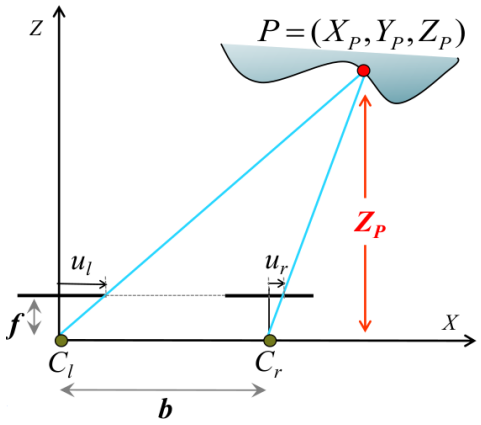


FIGURE 2.6: Stereo disparity illustrated [60].

Fig. 2.6 illustrates stereo disparity computation, where both cameras, with centers C_1 and C_2, observe the point P. The disparity can be computed intuitively, using similar triangles, and observing f and Z_p, and their counterparts u_l and X_p (or u_r and $X_p - b$), respectively.

With the goal to store the disparity and depth information independently from the camera pairs, their baseline and focal length, and to exploit the whole dynamic range of a depth map, one may employ a disparity normalization:

$$d_{norm} = \frac{d - d_{min}}{d_{max} - d_{min}} \cdot (2^n - 1), \tag{2.14}$$

where d_{norm} is a normalized disparity value, and $(2^n - 1)$ is the maximal binary value for representing the depth, *i.e.*, 255 in the case of $n = 8$-bit depth maps. Moreover, d_{min} is the disparity of the furthest object considered in the scene, and d_{max} is the disparity of the nearest object in the scene. Applying the equations:

$$\begin{aligned} d_{min} &= f \cdot \frac{b}{Z_{far}} \\ d_{max} &= f \cdot \frac{b}{Z_{near}} \end{aligned} \tag{2.15}$$

into Eq. (2.14) and simplifying it (where Z_{near} and Z_{far} represent z-distance to the nearest and the furthest object in the scene, we obtain:

$$d_{norm} = \frac{\frac{1}{Z} - \frac{1}{Z_{far}}}{\frac{1}{Z_{near}} - \frac{1}{Z_{far}}} \cdot (2^n - 1). \tag{2.16}$$

These normalized values are afterward used in the scope of MPEG, for different software (depth estimators, view synthesizers, TMIV, MV-HEVC, 3D-HEVC). For each of the tested video sequences, Z_{near} and Z_{far} values are given in the camera parameters, and the further computation is straightforward, given the Eq. (2.16).

2.4.3 Classical depth estimation approaches

This section gives an overview of the algorithms used by the two classical depth estimation approaches tested in this thesis: Depth Estimation Reference Software (DERS) [62, 63] and Immersive Video Depth Estimation (IVDE) [64], both used in the MPEG standardization for the development of the novel immersive video technologies. Most depth estimation algorithms rely on matching corresponding image regions or blocks in two or more cameras with slightly different positions to determine the disparity between corresponding texture pixels. Both described methods are constituted of the matching cost computation and the graph cuts energy minimization. Additionally, they perform temporal consistency enhancement procedures to ensure a better visual quality for virtual navigation and speed up the estimation process.

In the case of **DERS**, the matching cost $C_{ij}(l)$ represents the cost of a depth label l selection to a pixel (i, j) in the reference image R, which gives a corresponding pixel (u, v) in the synthesized target image T. It is computed using a modified sum of absolute difference (SAD) metric on the 3×3 blocks of pixels centered on given pixels (i, j) and (u, v). The number of reference images in the older DERS version was up to 4 (left, right, top, and bottom view), while the newer DERS version [55, 63] supports any number of input reference views. In order to ensure temporal coherence for videos (avoid flickering artifacts) and to speed up the depth estimation process, the temporal enhancement step can be activated, which prevents the need to recalculate matching errors for pixels that have been designated as "non-moving" pixels. Finally, instead of computing the depth map with a "winner-takes-all" approach - which would result in a local optimal depth estimation - the authors initialize a Markov random field graph with labels and costs for each pixel and use the graph cut and α-expansion methods to obtain a satisfactory global depth estimation and minimize the energy function [65, 66]:

$$E = \sum_{l} \left(E_{data}(l) + E_{smooth}(l) \right) \tag{2.17}$$

E_{data} is the data attachment term and is obtained by combining the camera baseline D between T and R, the matching cost $C_{ij}(l)$, and the reliability weight R_w:

$$E_{data}(l) = D \cdot R_w \cdot C_{ij}(l), \tag{2.18}$$

where R_w is a reliability map, whose goal is to enable the depth estimation in flat texture areas (spatial smoothness) by weighting the matching error based on the

color slopes (spatial gradient) S between adjacent pixels of the reference image (R_{th} is a reliability threshold):

$$R_w = \begin{cases} \frac{R_{th}}{0.3}, & \text{if } S < 0.3 \\ \frac{R_{th}}{S}, & \text{if } 0.3 < S < R_{th} \\ 1, & \text{if } S > R_{th} \end{cases}$$

Aside from the cost and the reliability map, the energy function is also influenced by a smoothing map S_w, whose goal is to give more importance to the labels with smaller depths (close objects) and avoid over-propagation to background area:

$$E_{smooth}(l) = \lambda \cdot S_w \cdot E_{ld}, \tag{2.19}$$

where E_{smooth} is a regularization term, λ is a smoothing coefficient, E_{ld} is a depth continuity error, *i.e.* label difference (giving more importance to close labels), and S_w is:

$$S_w = \begin{cases} 1, & \text{if } S < \rho \cdot S_{th} \\ \frac{\rho \cdot S_{th}}{S}, & \text{if } \rho \cdot S_{th} < S < S_{th} \\ \rho, & \text{if } S > S_{th} \end{cases}$$

where ρ is the second smoothing coefficient, and S_{th} is a smoothing threshold.

Finally, the depth estimation process ends when the identified best set of labels l gets converted to a normalized depth map.

IVDE, another tested approach, estimates depth for segments $s \in S_v$ instead of for each pixel individually, as shown in Fig. 2.7. The segment borders are calculated with SNIC method [67], and they are collocated with object boundaries, which helps to ensure the quality of computed depth maps.

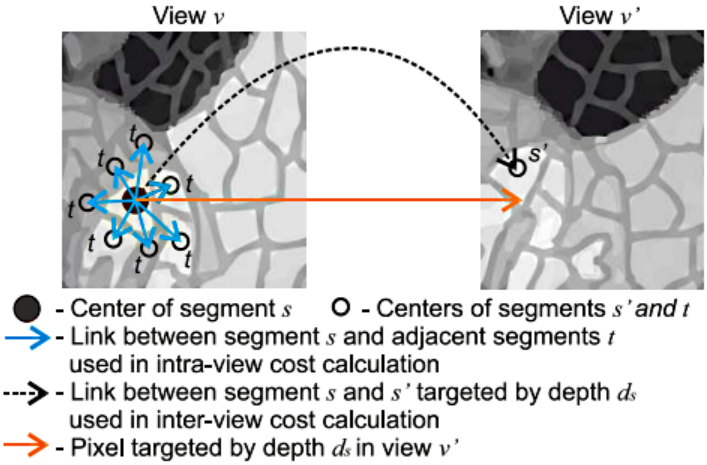


FIGURE 2.7: An example of IVDE intra-view discontinuity cost and inter-view matching cost on a segment s [64] ©2020 IEEE.

Moreover, the cost function has two different constituents: intra-view discontinuity cost $D_{s,t}$ with a smoothing regularization term, computed for each individual view, and inter-view matching cost $M_{s,s'}$, which is defined between the view $v \in V$ and its neighboring view $v' \in W_v$:

$$E(\underline{d}) = \sum_{v \in V} \sum_{s \in S_v} \left(\sum_{v' \in W_v} M_{s,s'}(d_s) + \sum_{t \in T_s} D_{s,t}(d_s, d_t) \right), \tag{2.20}$$

where $\underline{d}$ is a vector of depth values for all segments in all views, d_s is the depth of segment s which is currently being considered, $t \in T_s$ are segments neighboring to s, and d_t id the depth of segment t which is currently being considered. For a particular segment s, any depth value d_s can be selected, which can point at any pixel in the view v', and not just to the s' segment center (see Fig.2.7).

The intra-view discontinuity cost is calculated as:

$$D_{s,t}(d_s, d_t) = \beta \cdot |d_s - d_t|, \tag{2.21}$$

where β is a smoothing coefficient, calculated using the initial smoothing coefficient given by the user, and the L_1 distance between the segments s and t.

The inter-view matching cost is computed in the pixel domain, in a window A around the center of a given segment and the corresponding point in a neighboring view, using L_1 distance $\| \cdot \|_1$. First, we have:

$$m_{s,s'}(d_s) = \frac{1}{size(A)} \sum_{a \in A} \left\| [YC_bC_r]_{\mu_s + a} - [YC_bC_r]_{T[\mu_s] + a} \right\|_1, \tag{2.22}$$

where a is a point in window A, μ_s is the center of the segment s, $[YC_bC_r]_{\mu_s + a}$ is the vector of Y, C_b, C_r color components of the μ_s center of the segment s, $T[\cdot]$ is a 3D transform derived from intrinsic and extrinsic camera parameters, $[YC_bC_r]_{T[\mu_s] + a}$ is the vector of Y, C_b, C_r color components of the point in the view v' which is obtained with the 3D transformation from μ_s center of the segment s in the view v. Furthermore, the inter-view matching cost $M_{s,s'}$ is calculated as follows:

$$M_{s,s'}(d_s) = \begin{cases} min\{0, m_{s,s'}(d_s) - K\}, & \text{if } d_s = d_{s'} \\ 0, & \text{if } d_s \neq d_{s'} \end{cases}$$

where d_s is a currently considered depth of segment s in view v, while $d_{s'}$ is a currently considered depth of segment s' from view v'. Here, $K > 0$ and it represents a threshold for $m_{s,s'}$; if surpassed, it means that segments s and s' are different objects and the inter-view matching cost will not decrease the overall cost function $E(\underline{d})$. K is a value that models the noise in the images:

$$K \approx N_\sigma \cdot \sqrt{N_v} \cdot \sqrt{N_c} \cdot \sigma, \tag{2.23}$$

and equal to $K = 30$, as derived from: the standard deviation of typical noise $N_\sigma = 5$, of two views $N_v = 2$, three color components $N_c = 3$, and standard deviation σ of noise distribution existing in a single source (the value up to 2.5).

As opposed to DERS, the depth estimation is executed simultaneously for all the views, and the resulting depth maps are inter-view consistent. Consequently, this makes the technique robust to possible specular reflections and also attenuates the influence of other non-Lambertian reflections on the accuracy of depth maps. Furthermore, this method makes no assumptions about the position of the views, and the optical axes of the cameras do not need to be parallel, making this method more general and convenient for immersive video applications. The method also utilizes the temporal consistency enhancement procedure, which additionally speeds up the processing time. The graph cut method and α expansion are used to compute the minimum of the cost function, where each node corresponds to one segment.

2.4.4 Learning-based depth estimation approaches

This section details the two neural depth estimation approaches tested in the scope of this thesis, which go beyond MPEG standardization activities: Group-wise Correlation Stereo Network (GwcNet) [57] and Guided Aggregation Net for End-to-end Stereo Matching (GA-Net) [58]. Although not initially developed for view synthesis purposes, both methods were recognized in the literature as highly performing (at the time of our studies) and were evaluated in the context of depth image-based rendering for immersive video during our experiments. The methods perform stereo matching on a pair of rectified images and they are constituted of convolutional neural network (CNN) layers. Both GwcNet and GA-Net are tested with their pre-trained models, trained on the Scene Flow [68] (synthetic stereo video dataset with different scenes) and KITTI [69] (real-world stereo videos of road scenes) datasets.

GwcNet is a network that aims to create the cost volume through group-wise correlation and it relies on four parts: unary feature extraction, cost volume construction, 3D aggregation, and disparity prediction, as shown in Fig. 2.8.

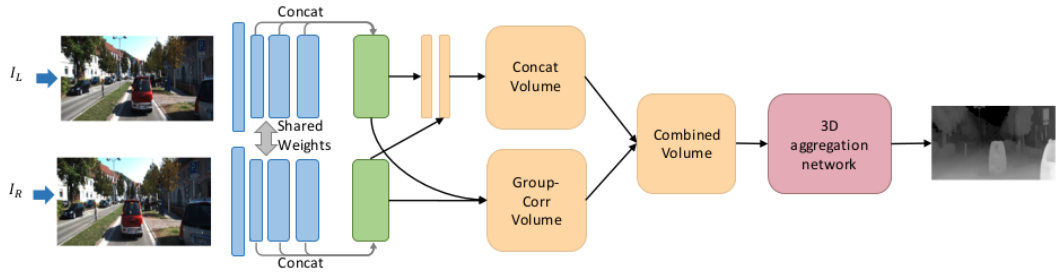


FIGURE 2.8: The high-level block scheme of GWCNet architecture [57] ©2019 IEEE.

Firstly, ResNet-based network is utilized to extract high-level features from rectified stereo images, separately for each view. The resulting feature maps from the last three layers of the ResNet-based network are combined into a unary feature map for each view. The next step involves creating the cost volume, which is formed by merging the concatenation volume and group-wise correlation volume. The concatenation volume is generated by merging the unary features for both views (which are first compressed with two convolutions), while the correlation volume is computed by performing correlation among groups of left and right features. Subsequently, the concatenation and correlation volumes are merged into a combined volume by concatenation, which is then used to predict the cost volume. Afterward, a 3D aggregation network, consisting of a module with convolutional layers followed by three stacked 3D hourglass encoder-decoder networks, predicts a refined cost volume. Finally, the 3D aggregation network is connected to four output modules that regress a disparity map.

In output modules (3D aggregation), two 3D convolutions compute a 1-channel 4D volume, which is subsequently upsampled and converted into a probability volume with softmax function $\sigma(\cdot)$ along the disparity dimension. Each pixel has the vector of length D_{max} which contains the probability p for all disparity levels. Next, the disparity $\tilde{d}$ estimation is given by the soft argmin function:

$$\tilde{d} = \sum_{k=0}^{D_{max}-1} k \cdot p_k, \tag{2.24}$$

where k and p_k stand for a potential disparity level and its corresponding probability. Four output modules predict disparity maps $\tilde{\boldsymbol{d}}_0, \tilde{\boldsymbol{d}}_1, \tilde{\boldsymbol{d}}_2, \tilde{\boldsymbol{d}}_3$. For disparity regression, the loss function is defined as follows:

$$\mathcal{L} = \sum_{i=0}^{i=3} \lambda_i \cdot Smooth_{L_1}(\tilde{\boldsymbol{d}}_i - \boldsymbol{d}^*), \tag{2.25}$$

where λ_i denotes the coefficients of the disparity prediction $\tilde{\boldsymbol{d}}_i$ and $\boldsymbol{d}^*$ denotes the ground truth disparity. The $Smooth_{L_1}$ is calculated as follows:

$$Smooth_{L_1}(x) = \begin{cases} 0.5x^2, & \text{if } |x| < 1 \\ |x| - 0.5, & \text{otherwise} \end{cases}$$

It is robust to the disparity discontinuities and has low sensitivity to noise or outliers.

GA-Net, the second tested deep learning approach, intends to replace the computationally costly 3D convolutions for cost volume regularization, by employing semi-global guided aggregation (SGA) and local guided aggregation (LGA) layers. Fig. 2.9 depicts its architecture which consists of: feature extraction (stacked hourglass CNN connected by concatenations), 4D cost volume aggregation, the guidance sub-network which produces cost aggregation weights, and disparity estimation.

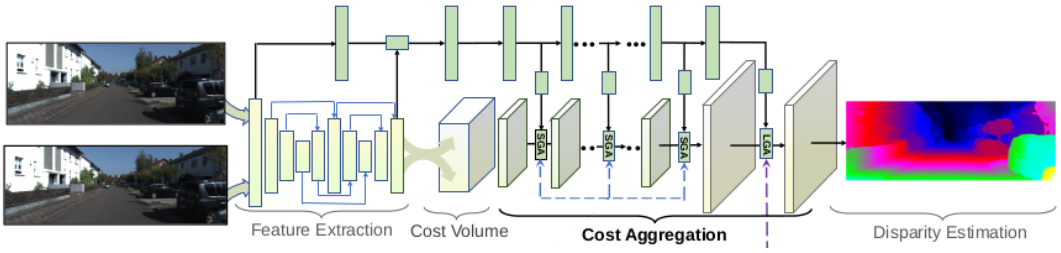


FIGURE 2.9: The high-level block scheme of GA-Net architecture [58]
©2019 IEEE.

A differentiable approximation of semi-global matching is used by SGA, and it aggregates the matching cost over the entire image in various directions, which allows for precise estimations in occluded areas or reflecting and textureless areas. LGA refines the thin structures and makes up for the downsampling that is done before the cost volume block.

For training the disparity regression, similarly, as for GwcNet, the loss function is defined as follows:

$$\mathcal{L}(\hat{d}, d) = \frac{1}{N} \sum_{n=1}^{N} Smooth(|\hat{d} - d|), \tag{2.26}$$

where disparity prediction $\hat{d}$ is the sum of each disparity candidate weighted by its probability (as given in Eq. (2.24)), $|\hat{d} - d|$ is the absolute error of predicted disparity, N is the number of valid pixels with ground truth values for training, and $Smooth$ function is defined as follows:

$$Smooth(x) = \begin{cases} 0.5x^2, & \text{if } x < 1, \\ x - 0.5, & \text{if } x \geq 1. \end{cases}$$

The presented learning-based methods have been evaluated on the Middlebury [70] and KITTI [71] benchmarks, showing good performance in terms of given stereo-matching metrics (root mean squared error, percentage of bad pixels, *etc.*). Such metrics evaluate the performance of stereo algorithms on various aspects such as accuracy and completeness. However, studies have demonstrated that such metrics are not strongly correlated with conventional metrics used to measure video quality, like PSNR and SSIM [72]. Therefore, our experiments in Chapter 5 investigate the performance of learning-based approaches in a particular context of view synthesis.

Our study has demonstrated that traditional depth estimation approaches are still more effective than learned-based methods. This is mainly because pre-trained models are vulnerable to domain shift issues between natural and computer-generated content. Additionally, learning-based methods often have poor computational efficiency when dealing with high-resolution content, which is typical in immersive video applications. On the other hand, learning-based approaches produce smoother depth maps, which are easier to compress. This property could be useful in a standard MIV scenario, when depth maps are compressed and transmitted, or in a hybrid-DSDE scenario.

The next section presents different view synthesis approaches, detailing the classical DIBR and learning-based IBR techniques.

2.5 Virtual view synthesis

While the multi-view video format presents opportunities for numerous future applications, efficiently acquiring, storing, transmitting, and processing this type of data still continues to pose a challenge. In this context, novel view synthesis methods are particularly important as they recreate the parallax effect of a 3D scene by enabling the creation of virtual viewpoints using a limited number of input images.

The different methods of rendering, along with their corresponding representations, are categorized into three groups: rendering without geometry, rendering using implicit geometry, and rendering using explicit geometry [73].

In our context, we deal with various camera setups with relatively large camera baselines and different camera projection formats (perspective/pinhole projection as well as equirectangular projection). Hence, within the scope of this thesis, we have employed view synthesis tools for a novel view generation that are versatile and recognized by the MPEG community and academia.

In the following sections, we will focus on the tools we utilized: first, we will provide an overview of depth image-based rendering principles and given solutions (methods with explicit geometry), and afterward, we will discuss image-based rendering with a closer look at the neural image-based solutions (methods with implicit geometry).

2.5.1 Depth image-based rendering

Depth image-based rendering (DIBR) is a technique that doesn't involve explicit reconstructing of the 3D scene geometry. Rather, it uses the corresponding depth map to warp the input color image. Advanced stereoscopic display processing technologies, such as autostereoscopic displays, benefit greatly from DIBR as it allows for the creation of a more immersive and realistic viewing experience. Below we will give a description of the DIBR process.

In Section 2.4.2, we have detailed the camera perspective projection, with the matrix $K[R\ T]$. Here, we give again the equation for the projection of the point P in the 3D space to the point p in the image plane of the camera:

$$z \begin{bmatrix} p \\ 1 \end{bmatrix} = K \begin{bmatrix} R & T \end{bmatrix} \begin{bmatrix} P \\ 1 \end{bmatrix}. \tag{2.27}$$

which is equivalent to:

$$z \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & c_u \\ 0 & f & c_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{2.28}$$

The Eq. (2.28) can be transformed, and homogeneous coordinates can be added, to obtain a more general equation with a projection matrix M [62]:

$$z \begin{bmatrix} u \\ v \\ 1 \\ 1/z \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{2.29}$$

which can be simplified to:

$$z \begin{bmatrix} u \\ v \\ 1 \\ 1/z \end{bmatrix} = M \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{2.30}$$

Following these equations, one can, given the depth value Z of the pixel and the input image, back-project the point P from the image plane to the world, using the inverse projection matrix M^{-1}:

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = zM^{-1} \begin{bmatrix} u \\ v \\ 1 \\ 1/z \end{bmatrix}. \tag{2.31}$$

Let us consider a pixel p_{in} from the input camera, with its projection matrix M_{in}, and its depth z_{in}, and the target pixel p_{tar} from the output camera, with its projection matrix M_{tar} and its depth z_{tar}. By using the Eq. (2.30) and Eq. (2.31) we can obtain a homography matrix H which projects the input pixel p_{in} to the target pixel p_{tar}:

$$z_{tar} p_{tar} = z_{in} M_{tar} M_{in}^{-1} p_{in}, \tag{2.32}$$

which can be simplified to:

$$z_{tar} p_{tar} = z_{in} H p_{in}, \tag{2.33}$$

where the matrix H has a dimension of 4×4 and is common for all depth values of different pixels. It can be calculated knowing input and target camera parameters.

The homography projection Eq. (2.33) should be applied to each pixel of the input image to generate a novel viewport. Nevertheless, not all the pixels in the novel viewport can be successfully generated with this technique. Some challenges still remain [74]:

- errors in depth maps, which can cause ghosting artifacts,
- differences in color among source views,
- occlusions, which happen when objects in source views occlude parts in the background which are visible from the target image,
- zoom or step into the scene, which can create cracks and dilation,
- increasing the baseline between the source cameras,
- rendering of non-Lambertian objects that exhibit transparency or specularity.

We will give a brief overview of the algorithms used for the novel view rendering with TMIV, and show how they are handling some of the challenges mentioned above. The two classical depth-image-based rendering approaches are Reference View Synthesis (RVS) [11, 75] and View Weighting Synthesizer (VWS) [24] software, both recognized by the MPEG standardization group for the development of MIV standard. Both methods are rendering target viewports from any number of input videos, support a large baseline, and handle the content with both perspective and equirectangular projections.

The first TMIV renderer alternative is derived from **RVS** [11]. It projects the source image points onto the target image points, using the triangles method [76]. A grid of triangles is formed by dividing each input image, with triangle vertices being the pixels that are adjacent to each other. These vertices are then reprojected to the new target image, triangles are deformed and rasterized. The projected pixels which are elongated are considered to lie on a disocclusion or a tangential surface, and they are discarded. The texture color and depth values in a triangle are computed via barycentric interpolation.

The pixel blending is independent of the rendering order. It is done in such a way as to prefer the closer views over views that are further away, to prefer foreground over background objects, and to penalize very elongated triangles, as given in the Eq. (2.34):

$$\begin{gathered} p_{blend} = \sum_i w\left(\gamma_i, d_i, s_i\right) \cdot p_i \\ w : (\gamma, d, s) \rightarrow \exp(-c_\gamma \cdot \gamma + c_d \cdot d - c_s \cdot s), \end{gathered} \tag{2.34}$$

where p_{blend} is the target pixel, computed from different contributions (pixels p_i). Moreover, w is an exponential function that maps the input variables given as a ray angle γ (between the input and target camera), reciprocal geometry d (the reciprocal of the depth value in the target view), and stretching s (the value of the triangle area in the target view with respect to the source view).

The **VWS** method is the TMIV renderer used in the scope of this thesis. It aims to alleviate the artifacts caused by the inconsistencies in source depth maps and to allow smoother transitions between viewports. To this end, VWS deploys visibility and shading steps.

The visibility step generates a target viewport depth map, using splat-based rasterization [24] in pixel reprojection to the target viewport. The selection of the depth pixels is done based on the carefully chosen weighting strategy, where weights depend on the distance between the input and target viewport position (tri-dimensional camera rigs) or the distance between the target viewport and the input view forward camera axis (linear or planar rigs). Moreover, the weight is updated during rendering, using TMIV pruning information, which gives additional importance to pixels that are reprojected on the pruned areas.

The shading step computes the color of the target viewport via the weighted blending of the input pixels. The input pixel weight is calculated depending on its consistency with the visibility map and its input view weight.

2.5.2 Neural image-based rendering

Image-based rendering methods are the ones that render novel views directly from available input images. To this end, they employ view interpolation, view morphing, or transfer methods. These techniques deal with the implicit scene geometry, and they obtain the 3D information of a scene by computing positional correspondences, such as point features [73].

Lately, there has been huge progress in the field of neural image-based rendering [77, 78]. These methods aim to render novel views of complex scenes captured by a sparse set of input views, and they can have different 3D scene representation formats. Furthermore, the field of neural image-based rendering has been transformed by the Neural Radiance Fields (NeRF) technique [79]. By utilizing multi-layer perceptrons (MLP) and positional encoding, it can represent a scene with remarkable accuracy. Here, the static scene is described as a continuous 5D function, which generates the radiance released in each direction (θ, ϕ) at every point (x, y, z) in space, and a density at each point. This density works as a differential opacity that governs the amount of radiance gathered by a ray that passes through (x, y, z). The model's ability to handle non-Lambertian effects while rendering new views is enabled by its view-dependent nature. Nevertheless, NeRF-based (implicit) models encode a scene into their own weights and require a lengthy optimization process to render an unfamiliar scene, which makes them impractical.

One of the recently developed methods, **IBRNet** [73], tested in the scope of this thesis, is a state-of-the-art, robust neural radiance field method that generalizes well to novel scenes. Its ability to generalize is a desired characteristic for an immersive video renderer in the transmission scenario. IBRNet builds both on traditional image-based rendering and on the NeRF approach [79], generating a continuous and differentiable scene radiance field. Its end-to-end design prioritizes the optimization of synthesis quality, outperforming recent one-shot synthesis approaches, like [77], while still having good quality synthesis results as compared to single-scene inference approaches.

IBRNet (Fig. 2.10) first identifies a set of neighboring source views to a target and extracts their image features. Then, for each camera ray in the target view, it computes colors and densities for a set of samples along the casted ray. Finally, colors and densities along that ray are accumulated to render the color of the target pixel. In the course of training, the mean squared error between the ground truth pixel color and the rendered pixel color is minimized.

More specifically, let us observe a point $\mathbf{x} \in \mathbb{R}^3$, with unit-length viewing direction $\mathbf{d} \in \mathbb{R}^3$, which lays on a ray $\mathbf{r} \in \mathbb{R}^3$. The point $\mathbf{x}$ is reprojected on N selected source views, which yields the corresponding obtained colors $\{C_i\}_{i=1}^N \in [0,1]^3$ and image features $\{\mathbf{f}_i\}_{i=1}^N \in \mathbb{R}^d$. The extracted image colors, and features, with directions, are given to an MLP architecture, with the goal to aggregate local and global information and obtain features that are multi-view aware.

Subsequently, instead of predicting the density σ straight from the density feature $\mathbf{f}_\sigma$, a ray transformer module is used. It takes M different samples on a ray $(f_\sigma(\mathbf{x}_i), ..., f_\sigma(\mathbf{x}_M))$, ordered from near to far, and passes them to a positional encoding and multi-head self-attention module that creates the final density value σ. The

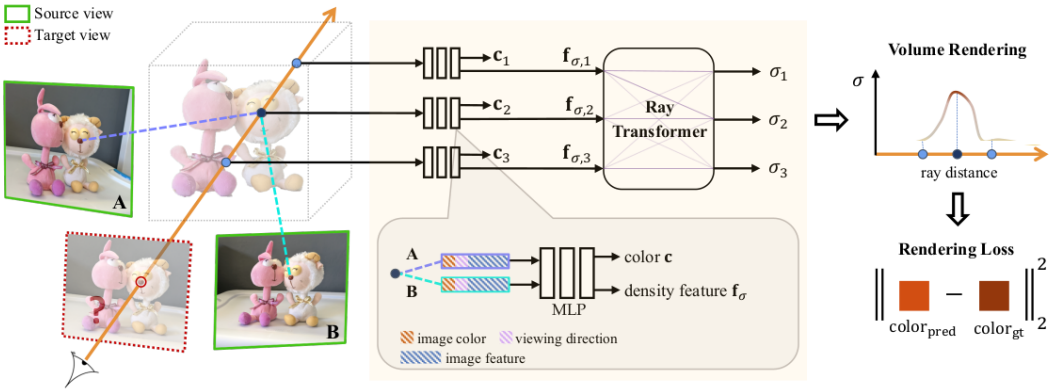


FIGURE 2.10: An overview on IBRNet pipeline [73] ©2021 IEEE.

described module enables the samples to attend to each other along a given ray, consequently improving geometric reasoning and density predictions.

To produce a color at a particular 5D point, this method predicts blending weights for the image colors $\{C_i\}_{i=1}^N$ in the source views that are associated with that point. For each source color C_i, the features are concatenated with the direction relative to the source view direction, which is the difference $\Delta\mathbf{d}_i = \mathbf{d} - \mathbf{d}_i$, with $\mathbf{d}$ being the view direction of the ray $\mathbf{r}$. These concatenated features are passed to a neural network that gives a blending weight w_i^c associated with each color. Finally, the color for the 5D point is blended with a soft-argmax operator:

$$\mathbf{c} = \sum_{i=1}^{N} \frac{C_i \cdot \exp w_i^c}{\sum_{j=1}^{N} \exp(w_j^c)}. \tag{2.35}$$

The Eq. (2.35) has been defined for a continuous 5D query location $(\mathbf{x}, \mathbf{d})$, yielding a color $\mathbf{c}$ and a density σ. To render the color along the ray $\mathbf{r}$, let us take M samples on the ray, sorted along the ray with ascending depths. The colors $\mathbf{c}_k$ are then accumulated and modulated by their densities σ_k:

$$\tilde{C}(\mathbf{r}) = \sum_{k=1}^{M} T_k \cdot (1 - \exp(-\sigma_k)) \cdot \mathbf{c}_k, \tag{2.36}$$

$$\text{where } T_k = \exp\left(-\sum_{j}^{k-1} \sigma_j\right). \tag{2.37}$$

The loss function is constructed by combining the outputs of two models, *coarse* and *fine*. They have an identical architecture, but operate on different sample inputs: the *coarse* yields a prediction on sample locations that are at the equidistant disparity, while the *fine* gets samples on the ray which are expected to be placed at the areas relevant for rendering. The loss function is then defined as follows:

$$\mathcal{L} = \sum_{\mathbf{r}\in R} \left[\left\| \tilde{C}_{\text{coarse}}(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 + \left\| \tilde{C}_{\text{fine}}(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 \right], \tag{2.38}$$

with R being the set of rays in a training batch.

The presented classical DIBR methods are fast and robust, but limited in their

ability to handle non-Lambertian content, due to the invalidity of the linear hypothesis of pixel displacement [75]. On the other hand, the benefits of neural IBR approaches, such as IBRNet, are their good rendering performance for very complex content, robustness, and generalization capabilities. Their drawback is the rendering time needed for high-resolution content, which is the usual case for immersive video applications.

2.6 Conclusion

This chapter has introduced immersive video representations and their coding solutions, with a particular overview of the general principles of video coding for the simulcast approach, and multi-view and 3D video coding solutions. Their drawbacks have been discussed and novel volumetric coding solutions were presented. The non-normative parts of the immersive video pipeline, depth estimation, and view synthesis are discussed. The presented algorithms will be used in the following Chapters, to evaluate the proposals presented in this thesis.

The next chapter will introduce the MPEG immersive video standard, its reference software, the notion of decoder-side depth estimation, and MPEG common test conditions, and it will give the comparative performance of different MIV profiles.

Chapter 3

MPEG Immersive Video

Since this thesis was mainly steered by the development of the MIV standard, and its Geometry Absent profile, in this chapter we give a comprehensive review of the most recent version of the Test Model for MPEG Immersive Video (TMIV), which serves as the benchmark software for the MIV standard. The TMIV codec pre-processes the source content, which involves converting input videos into atlases, that are essentially a set of non-redundant multi-view data. This chapter provides a detailed description of the TMIV encoding and decoding processes. In addition, we outline the testing conditions, including the datasets, software, and quality metrics utilized to evaluate the proposals. Finally, we analyze the performance of various MIV anchors.

3.1 Introduction

The immersive video systems described in the previous chapter were deemed inefficient and not versatile in compressing the MVD volumetric data. Hence, the need for a new standard emerged, due to the increasing demand for high-quality immersive experiences. This has led to a collective academic and industry effort to support immersive media access and delivery, culminating in the ISO/IEC 23090-12 MPEG Immersive Video (MIV) standard [23], which represents a critical milestone.

Firstly, a Call for Proposals for the coding of 3DoF+ videos, with the aim to enable user's head movements within a limited space, was issued in January 2019 [80]. The underlying technique for MIV involves the use of a 2D video codec for the compression of the transformed data, packed into atlases. Another standard for volumetric media, the ISO/IEC 23090-5 Video-based Point Cloud Compression (V-PCC), also uses video codecs to compress its data, although it has a different approach for the conversion of the 3D volumetric data into 2D content for compression. Thus, during the 129$^{\text{th}}$ MPEG meeting, the fourth working draft of MIV was adjusted to use ISO/IEC 23090-5 Video-based Point Cloud Compression (V-PCC) as a normative reference for various aspects such as definitions, terms, syntax, semantics, and decoding processes. The alignment process was completed during the 130$^{\text{th}}$ MPEG meeting by reorganizing Part 5 into a common specification called Visual Volumetric Video-based Coding (V3C), with annex H being the Video-based Point Cloud Compression (V-PCC) standard. Furthermore, V3C offers extension mechanisms for V-PCC and MIV. This standard alignment resulted in a change of the input data's nomenclature, which is the reason why in this thesis, we will interchangeably use the old terms labeled as *texture* and *depth* with newly adopted terms named *attribute* and *geometry*, respectively. Finally, the first edition of the MIV standard was released at the 135$^{\text{th}}$ MPEG meeting in July 2021 [23, 81]. The second edition of the standard is currently in development.

The MIV standard includes various profiles, to accommodate different input formats, bandwidths, and client decoding resources [82]. The MIV Main profile is used for MVD data, where the attribute is the texture, and the geometry is the depth information. It represents the occupancy of the atlas directly embedded in the depths, where occupancy indicates if a pixel in the atlas is valid or not (pruned). The MIV Extended profile allows the use of not only the MVD data but may separate the occupancy from geometry and activate an additional, transparency attribute. This enables a more advanced synthesis, which may be, *e.g.,* based on the segmentation of different objects using their materials or reflectance properties. Furthermore, the MIV Extended - restricted subprofile enables the coding and delivery of a multiplane image (MPI) format [83], a format known for its rendering efficiency, by transmitting attribute and transparency. Finally, the MIV Geometry Absent (GA) profile is created to transmit solely the textures, and avoid the depth (geometry) transmission. Thus, this profile allows the cloud-based or decoder-side depth estimation, at the client side.

This thesis is mainly relying on the notions of the *decoder-side depth estimation* (DSDE) and its encoding scheme, enabled as the MIV GA profile. The DSDE system offers several advantages compared to the other MIV profiles. Firstly, DSDE saves a significant amount of bitrate since the coded depth maps can represent more than half of the total bitstream in the MIV Main profile, particularly at low bitrates [26]. Additionally, it saves half of the originally used pixel rate, which can be instead utilized for encoding textures with 2D codecs designed and optimized for texture content. While 3D-HEVC could compress depths efficiently, this is not the case for video-based solutions like MIV. Therefore, when we encode and transmit solely the textures and recover the depths at the client (DSDE mode), we improve the quality of light field reconstruction. Although the computational complexity of depth estimation can be considered a drawback of the DSDE system, there have been multiple works proposing more efficient methods for the depth estimation at the decoder side [84, 85].

This chapter gives a detailed overview of the latest version of the Test Model for MPEG Immersive Video (TMIV 15), which is the reference software for the MIV standard. Since this thesis has been carried out from the very start of the MIV standardization process, the differences between the versions of TMIV used in the thesis experiments are also given below. Moreover, this chapter presents the test conditions (datasets, software, and quality metrics) followed in the evaluation of our proposals, and the performance of different MIV anchors.

3.2 Test Model for MPEG Immersive Video

In brief, the TMIV encoder pre-processes the source material, transforming the input videos into atlases. Atlases are a collection of data where most of the redundancies are removed. Selected important information is presented as different patches in the atlases, as shown in Fig. 3.1. TMIV encoder also generates the MIV bitstream, with the information about the cameras and patches.

Particularly, the TMIV encoder is a non-normative implementation of MIV which operates on the uncompressed source views and their camera parameters, selects some of the parameters automatically (atlas frame size, number of atlases), preprocesses the texture and depth map videos, and packs them into texture and depth atlases. Moreover, it produces the MIV metadata, containing the information required to decode the atlas video streams. Subsequently, the 2D video encoder compresses

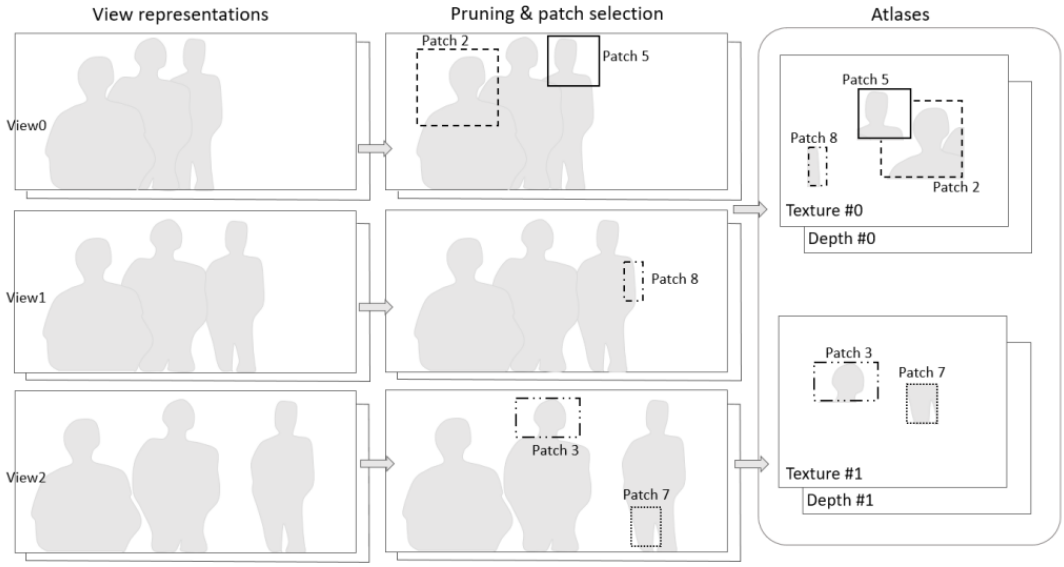


FIGURE 3.1: Input data processing with TMIV encoder [24].

the atlases independently of TMIV, resulting in the production of atlas video sub-bitstreams. A high-level block scheme of the encoding process is shown in Fig 3.2. For the sake of simplicity, here we show just the texture and depth components (without additional possible inputs such as transparency).

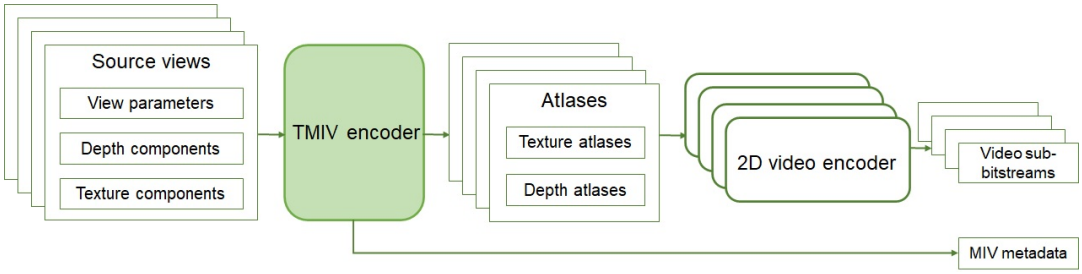


FIGURE 3.2: A high-level block scheme of the immersive video coding paradigm with MIV: encoding.

TMIV decoder reconstructs the pruned patches and recovers their information (ID, original view position, *etc.*), which is needed for subsequent rendering. This process is shown in Fig. 3.3.

More specifically, the TMIV decoder is a reference implementation of a normative process that decodes the MIV metadata, while the 2D video decoder decodes the atlas video sub-bitstreams. Afterward, the decoded atlases and MIV metadata are used to render the target viewport, with a non-normative rendering technique, such as RVS [11]. A high-level block scheme of the decoding process is shown in Fig 3.4.

Section 3.2.1 details the TMIV encoding process and Section 3.2.2 describes the TMIV decoding and rendering processes.

3.2.1 TMIV encoder

The TMIV encoder can encode the source views in one or multiple subsets (groups), where the number of groups is a user's choice. The purpose of view grouping is to identify bundles of views and individually analyze these bundles during the

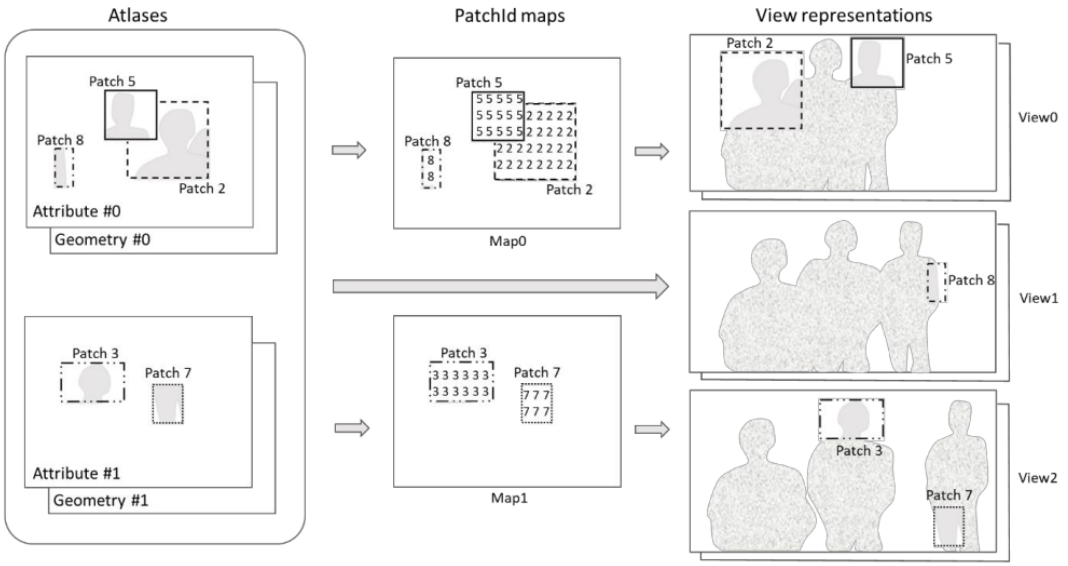


FIGURE 3.3: Decoding and reconstruction of TMIV pruned data [24].

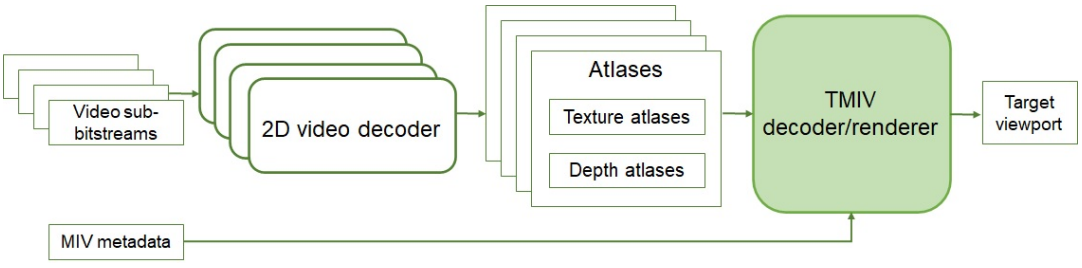


FIGURE 3.4: A high-level block scheme of the immersive video coding paradigm with MIV: decoding.

single-group encoding stage. Grouping is beneficial for preserving coherent non-redundant regions of the scene because it enforces the sampling (during pruning) to be done on the neighboring views. The TMIV encoder consists of a group-based encoder, which is always invoked at the beginning of the encoding process (independently of the number of groups), and a single-group encoder, which encodes each group individually (see Fig. 3.5).

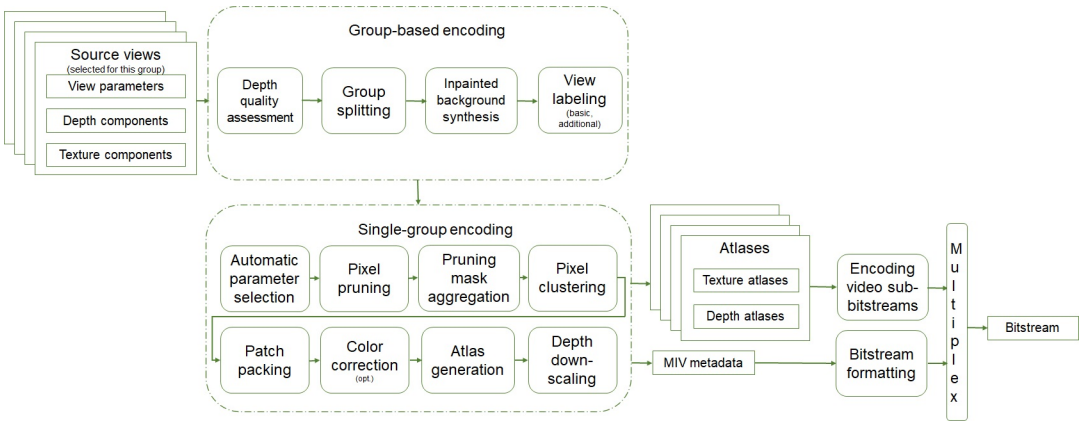


FIGURE 3.5: An overview of the TMIV encoding pipeline, with group-based and single-group encoding.

Group-based encoding

During the first encoding phase, called group-based encoding, the quality of depth maps is assessed for each source view. This assessment is done to verify if the level of inter-view consistency among the depth maps is sufficient. The depth map of each source view is reprojected to the others and the value of a depth pixel of the source view is compared with the depth value of the corresponding 3×3 pixel neighborhood in the target view. If the difference is higher than a defined threshold, the source pixel is considered inconsistent, and when the total amount of inconsistent pixels is higher than a given threshold, the depth map quality is labeled as low. If the depth map quality is low, the renderer will not use it in the intermediate view synthesis process. Instead, it will warp the texture pixels from the closest source views to fill the holes.

Then, if the user requests more than one group, the source views are organized into multiple separate sub-groups. The grouping process is done by determining the key positions in the scene using the range and distance of cameras, by first finding the dominant axis and the furthest view positions in the pool of views, and then gathering the closest views to the key positions, for each group. The number of key positions is equal to the number of groups.

Afterward, the synthesized inpainted background texture and depth view is prepared, which is to be used instead of the inpainted data at the decoder side, and it is additionally packed into the atlas. This process is designed to create the synthetic data that is "hidden" from the non-pruned source views, which consequently reduces the complexity of the inpainting at the decoder. It is utilized to fill the pixels that are lacking after intermediate view synthesis because at the decoder side, after view reconstruction, only partial views are available due to pre-processing in the TMIV encoder. The background view is synthesized from all source views, by rendering the background over the foreground. The missing data are inpainted from neighboring areas using the push-pull inpainter [86].

Finally, some of the source views are labeled as *basic views*, and the rest of them are labeled as *additional views*. Basic views are the views that are transmitted fully, while additional views undergo detailed processing in the single-group encoding step. However, there are two possible operating modes in TMIV, one which enables the use of both basic and additional views, and another one, which enables having only basic views as the output that is packed into atlases. Both of these modes are used in the evaluation of the MIV's performance, as will be detailed in Section 3.4. The labeling method is based on the partitioning around medoids (k-medoids) algorithm [87], where the basic views are k medoids among the source views, and the repulsion/attraction cost function is formed based on the distance between view positions.

Single-group encoding

The single-group encoding phase (see Fig. 3.5) is done on each group separately, after all the steps in the group-based encoding phase are done. Therefore, it will be done either on all the source views at once (if the number of groups is one) or independently for each grouped subset of views.

In the beginning, the automatic calculation of the number of atlases and the atlas frame size is done, according to the following constraints: the maximum size of a luma picture, the maximum luma sample rate, and the total number of decoder instantiations. These constraints are set by the MPEG group, with careful consideration of potential applications [20].

The next step is atlas construction, which mainly consists of pruning, patch clustering, and packing. Basic views are entirely packed into atlases, while additional views are processed, and their redundancies are removed before the packing step (see Fig. 3.1).

Pruning is an essential element of TMIV, because it reduces the inter-view correlation of the source multi-view set. As a reminder, MIV does not exploit the inter-view redundancies in the same way as MV-HEVC does, neither the inter-component redundancies as 3D-HEVC does. Therefore, the pruning process is carefully designed to remove redundancies among the views while maintaining realistic complexity. It determines which pixels from the additional views are already present in basic or other additional views, and thus, are redundant. A pruning oriented graph is introduced (as given in Fig. 3.6), which describes the hierarchy of the source views where the pruning is done. The basic views are the roots of this graph, while the additional views are assigned to the child nodes, in the order of decreasing number of preserved pixels. To evaluate if a pixel may be pruned, the pruner considers three criteria:

1. The views farther up the hierarchy should be used to synthesize the current pixel, as shown in Fig. 3.6. The pixel should be preserved in view of the parent node and pruned in view of the child node.

2. The difference between synthesized and source depth values should not exceed a given threshold, which is obtained experimentally and is given in the encoding configuration file.

3. The minimum difference between the luminance value of a synthesized pixel and luma of all pixels within a corresponding 3×3 source block should be smaller than a given threshold for luminance. This threshold as well is computed experimentally, and it adapts to the noise level of each sequence.

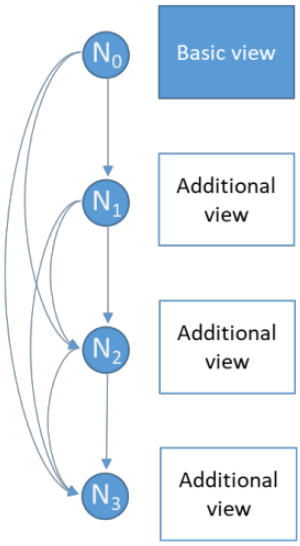


FIGURE 3.6: Pruning graph [24].

Subsequently, a second-pass pruning based on global color matching is employed to restore some of the initially pruned pixels that were pruned due to depth imperfections or illumination changes among different source textures [88]. First, pixel-by-pixel color differences are computed between the original view and the warped, pruned view at the same viewpoint, as shown in Fig. 3.7. Then, a fitting function is obtained with the least squares method, and it models the calculated color differences. Finally, the pixels that are outliers from the fitting function (outside a given interval) are restored, while the others remain to be pruned.

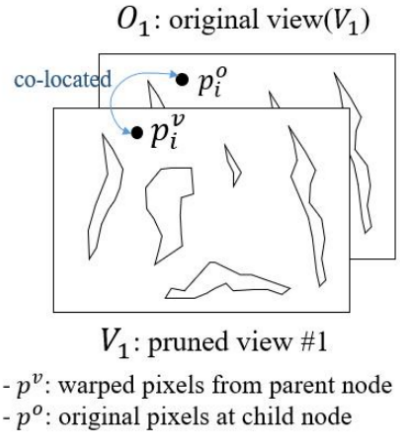
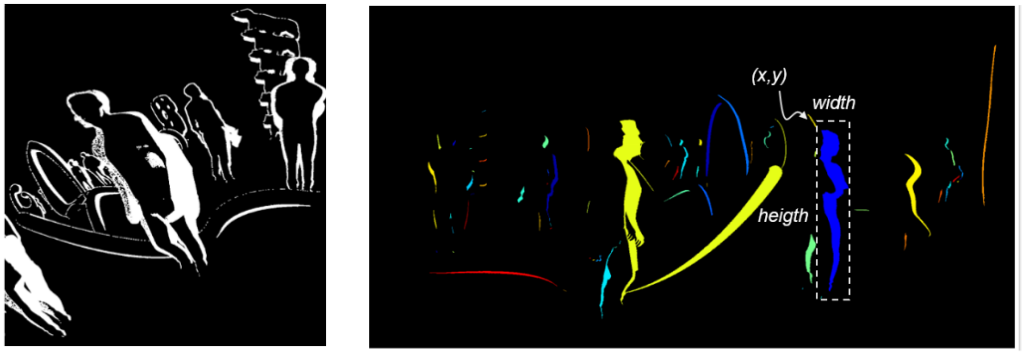


FIGURE 3.7: Second-pass pruning [24].

For each view, a mask of active pixels (pixels that were not pruned) is created. Temporal consistency is achieved with the mask aggregation process. Mask aggregation is a method of aggregating masks of active pixels frame by frame, and this is done for each intra period, while clustering is identifying sets of connected active pixels. Mask aggregation, with black pixels as pruned, and clustering process, where different clusters are in different colors, are shown in Fig. 3.8. Moreover, merging and splitting of the clusters in the pruning mask is done to distill the patches. These patches are then consecutively packed into the atlases, with possible rotations and



(A) Aggregated mask. (B) Clustering.

FIGURE 3.8: An example of mask aggregation and clustering [24].

flipping, to improve the coding efficiency. Packing is done with an algorithm based on MaxRect [89], which allows for overlapping slices. Larger patches are packed before the smaller ones, to ensure graceful degradation in the event that not all patches can fit into the given atlas space [90]. After packing, the patch colors are modified such that its new average value becomes a neutral color, *i.e.,* the mean value in a given bit range, since it has been shown that this improves the atlas compression performance with a 2D video codec.

The optional process of color correction aligns the color characteristics of the source views among each other. Moreover, depth and occupancy atlases can be downscaled, where depth is scaled by a factor of 2 with a max-pooling 2×2 filter. This allows for pixel rate savings and a better depth map encoding quality on a fixed bitrate (using lower QP values as compared to the non-scaled depths).

After atlas construction, the texture atlas and depth atlas video streams are encoded with an arbitrary 2D video encoder. All camera and atlas metadata parameters are formatted as a V3C sample stream with MIV extensions [91]. Finally, these bitstreams are multiplexed into a single MIV bitstream.

3.2.2 TMIV decoder and renderer

The pipeline of a TMIV decoder starts with a client's request for the desired viewport, by giving the viewport position and orientation. Then, the sub-bitstreams are demultiplexed, the MIV bitstream is parsed by the MIV decoder, and the video bitstreams are decoded by a 2D video decoder. Afterward, the frames are unpacked from the atlases, and the blocks in the frame are organized into a patch map which guides the recovery of basic and additional views, as depicted in Fig. 3.3. This process is followed by a non-normative (optional) depth estimation, in the case of decoder-side depth estimation, and finally, a non-normative rendering of the target viewport.

The process of rendering has multiple steps. First, the culling of the patches takes place to remove the patches that have no overlap with the target view, therefore reducing the complexity of the view synthesis. Furthermore, the occupancy frame is reconstructed, showing whether the pixel is occupied or non-occupied. Subsequently, since the patch colors are modified at the encoder, their mean value is restored at the decoder. After these steps, the additional (pruned) views are reconstructed, by mapping the texture and depth patches from the atlases back to their corresponding views.

In the case of the geometry absent profile, the depth maps are not transmitted; therefore, the estimation of depth maps may be done on the reconstructed textures, using a suitable depth estimation software, which is not integrated into TMIV. The reference software used for depth estimation is Immersive Video Depth Estimation (IVDE) [64] software, as decided by the MPEG experts. Previously, Depth Estimation Reference Software (DERS) [62, 63] was used as the reference software, and it is still a possible alternative for evaluation purposes. Both software are detailed in Section 2.4.3.

The view synthesis is implemented in TMIV with two different approaches. The MIV CTC defines the View Weighting Synthesizer (VWS) as the reference synthesizer. The second alternative is derived from the Reference View Synthesizer (RVS) [11]. These software are described in Section 2.5.1.

3.2.3 A word on different TMIV versions

The work on this thesis started at the same time as the MIV standardization. In our experiments, we used available test model versions, which have been improving over time. The most important version difference affects the results in Chapter 6, due to differences in the TMIV pruning process. Section 6.2 shows the results obtained with TMIV4 [92], which uses the pruning that depends only on the depths (the second criterion for pruning detailed in Section 3.2.1). However, in Section 6.3 we use TMIV7 [93]. Here, the pruning process has evolved into the more advanced version, which depends on depths and textures, and includes second-pass pruning (as detailed in Section 3.2.1). We emphasize this directly in Chapter 6, as well.

3.3 Test conditions

In this section, we present the MIV common test conditions (CTC) followed in the scope of this thesis [94]. The CTC defines the pipeline and procedure for the fair evaluation of proposals, and describes the MIV anchors, test sequences, and evaluation methodology. Similar to the TMIV reference software, the CTC was evolving over time during MIV standardization. In our experiments, we follow it closely, with some exceptions detailed in each chapter.

3.3.1 Datasets

The test material consists of both natural and computer-generated content, created with perspective and omnidirectional cameras. Therefore, the views have an omnidirectional format (equirectangular projection, ERP) or linear perspective format (linear perspective projection, LPP). In the case of natural content, depth maps are estimated, with some additional post-processing steps, while in the case of computer-generated content, they are produced using mathematical models of the generated 3D scene. In the CTC, the material is separated into mandatory and optional sequences, where optional sequences are more challenging but not compulsory for evaluation. A summary of the test sequences' characteristics is presented in Tab. 3.1. Only Chapter 4 gives an evaluation on both omnidirectional and perspective content, while Chapters 5, 6, and 7 and Appendix A focus on the perspective content. In all our experiments, we use 17 video frames for evaluation.

TABLE 3.1: Test sequences.

Sequence	Type	Format	Resolution	NumViews
Mandatory content				
Painter	NC	LPP	2048×1088	16
Frog	NC	LPP	1920×1080	13
Carpark	NC	LPP	1920×1088	9
ClassroomVideo	CG	ERP	4096×2048	15
Museum	CG	ERP	2048×2048	24
Fan	CG	LPP	1920×1080	15
Kitchen	CG	LPP	1920×1080	25
Chess	CG	ERP	2048×2048	10
Group	CG	LPP	1920×1080	21
Optional content				
Fencing	NC	LPP	1920×1080	10
Hall	NC	LPP	1920×1088	9
Street	NC	LPP	1920×1088	9
Mirror	NC	LPP	1920×1080	15
ChessPieces	CG	ERP	2048×2048	10
Hijack	CG	ERP	4096×2048	10
Cadillac	CG	LPP	1920×1080	15
Shaman	CG	LPP	1920×1080	25

3.3.2 MIV anchors

The CTC defines three different anchors for coding with TMIV [94]. The first one is called the *MIV anchor*. This anchor processes the source views as detailed in Section 3.2.1, with the goal to obtain the compact representation of the scene packed in the atlases, where some of the views are fully sent, while others are pruned. Secondly, the CTC introduces the *MIV view anchor*, which automatically decides which subset of views (textures and depth maps) to fully pack and transmit, without further processing of the source views. Finally, the *MIV decoder-side depth-estimating (DSDE) anchor*, is introduced, which automatically decides which subset of texture views to fully pack and transmit, whereas the depth maps are estimated using IVDE software at the decoder side before commencing the rendering process.

3.3.3 Software settings

The test sequences are processed with TMIV (versions explained in Section 3.2.3) and compressed with a legacy video codec. The CTC [94] specifies VVenC [95, 96] as an encoder, a fast implementation of VVC [97]. The VVenC encoder includes five predefined preset configurations: "slower", "slow", "medium", "fast", and "faster". Each of these configurations offers a different compromise of compression quality with respect to complexity (encoding speed). The "slow" preset has been selected by the MIV group as the best compromise, and the encoder is utilized with the random access configuration [98]. After the video transmission, the MIV metadata and the video bitstreams are decoded using TMIV decoder and VVdeC [99]. The target viewport from is rendered using all available decoded atlases, on the same positions as source cameras, to facilitate the objective quality evaluation.

The quantization parameters $\mathbf{QP_T}$ and $\mathbf{QP_D}$ for the compression of texture videos are chosen to cover the medium and low bitrate range, from 5 Mbps to 50 Mbps, and they are sequence-dependent. Then, the mapping to the quantization parameter $\mathbf{QP_D}$ [100] for the depth is done with the equation (3.1), applied to five rate points:

$$\forall i \in \{1,2,3,4,5\}\,, \quad \mathrm{QP_D}(i) = \max\left(1, \left[-14.2 + 0.8 \cdot \mathrm{QP_T}(i)\right]\right). \tag{3.1}$$

It is important to mention that, in older versions of the CTC [101], which we follow in Chapter 6, the video codec of choice is HM16.16 [102], an implementation of HEVC. The set of (QP_T, QP_D) pairs is: $\{(22,4),(27,7),(32,11),(37,15),(42,20)\}$. In this case, the QP parameters are not sequence dependent.

The pixel rate constraints are the following [20, 94]:

- The combined luma sample rate in all decoders does not surpass 1 069 547 520 samples per second (according to HEVC Main 10 profile level 5.2).
- Maximum luma picture size of each coded video does not surpass 8 912 896 pixels (*e.g.* 4096 × 2048).
- The maximum number of decoder instantiations is four.

All anchors comply with the low pixel rate constraint, which is fixed at 1.070 gigapixels per second. TMIV automatically computes the atlas frame sizes based on given pixel rate constraints.

3.3.4 Quality evaluation

The objective evaluation is mostly done by computing the Bjøntegaard delta (BD) rates [103, 104] in terms of different metrics. BD-rate shows the average bit rate savings for the same video quality (given with a chosen metric), computed between two rate-distortion curves. We use Y-Peak Signal-to-Noise Ratio (Y-PSNR) of synthesized views in comparison to the uncoded source views, which represents the value of the PSNR computed on the luminance channel of the video. Another metric is Immersive Video PSNR (IV-PSNR) [105, 106], which is a PSNR-based metric adapted for immersive video applications that alleviates the influence of pixel shifts caused by projection errors and the influence of global color differences among the different input views. Another group of metrics is based on Structural Similarity Image Assessment (SSIM) [107], which focus on the extraction of structural information from the images. Metric called Multiscale SSIM (MS-SSIM) [108] compares contrast and structure across several scales of the tested images. Additionally, we use learning-based methods such as Video Multimethod Assessment Fusion (VMAF) [109] and Learned Perceptual Image Patch Similarity (LPIPS) [110].

3.4 Performance of MIV anchors

This section gives a comprehensive comparison of three anchors and their performance: MIV, MIV view, and MIV DSDE.

3.4.1 Results and discussion

The MIV view and MIV DSDE anchors are compared to the MIV anchor in terms of BD-rate and video encoding runtime (which is given as a percentage of the MIV anchor runtime) in Tab. 3.2. Negative values of BD-rate mean that the tested anchor (MIV DSDE or MIV view) performs better than the reference MIV anchor, while positive values indicate that the tested anchor performs worse than the reference. In the case of "---" and "+++" there was not enough overlap to compute BD-rate, where "---" indicates gain, while "+++" indicates loss, compared to the MIV anchor.

We notice that the MIV view performs significantly better than the MIV anchor in several cases. The Hall sequence is an interesting exception, as it is the only sequence

TABLE 3.2: Comparison of the MIV view (View) and MIV DSDE (DSDE) anchors with respect to the MIV anchor.

Mandatory content										
	High BD-Rate PSNR [%]		Low BD-Rate PSNR [%]		High BD-Rate IV-PSNR [%]		Low BD-Rate IV-PSNR [%]		Video encoding [%]	
Sequence	View	DSDE	View	DSDE	View	DSDE	View	DSDE	View	DSDE
Painter (NC)	-33.9	-63.2	-31.9	-60.5	-18.6	-33.1	-24.4	-43.0	66.8	80.7
Frog (NC)	-33.5	-67.5	-29.7	-57.5	-9.3	-48.4	-17.0	-48.9	69.2	81.9
Carpark (NC)	31.5	2.1	13.2	-21.1	42.0	18.6	18.1	-10.0	82.6	81.7
Class.Video (CG)	4.0	+++	-20.4	50.3	4.3	158.3	-17.5	4.7	83.1	123.1
Museum (CG)	36.6	320.4	-4.5	73.8	44.4	113.8	0.5	25.1	58.4	88.3
Fan (CG)	-18.9	---	-15.6	---	-11.0	-69.6	-9.0	-77.5	71.1	131.4
Kitchen (CG)	75.8	-8.8	20.6	-6.5	206.9	105.9	55.0	51.7	63.6	92.2
Chess (CG)	+++	+++	+++	+++	+++	+++	+++	+++	62.5	62.5
Group (CG)	-47.2	+++	-42.4	+++	-11.6	+++	-20.2	+++	54.1	100.2
Optional content										
Fencing (NC)	28.4	67.4	-10.8	-5.4	16.5	25.3	-11.1	-25.7	57.0	82.0
Hall (NC)	1070.3	1060.2	74.6	377.8	119.3	429.3	33.7	177.8	82.9	108.6
Street (NC)	-40.1	-66.0	-17.1	-48.1	7.0	-33.2	11.9	-30.9	74.8	81.0
Mirror (NC)	48.9	-9.5	5.1	-31.2	100.6	16.7	13.2	-16.9	61.2	69.7
ChessPieces (CG)	+++	+++	+++	+++	+++	+++	+++	+++	55.1	66.8
Hijack (CG)	+++	+++	47.1	+++	+++	+++	66.4	+++	56.3	86.4
Cadillac (CG)	-30.6	-58.6	-39.5	-67.8	3.3	-23.9	-30.3	-50.4	55.4	54.8

with significant global camera motion. In such a case, the MIV anchor performs better, as temporal variations may be efficiently included in the atlases. Moreover, when inter-view consistent depth maps are present, as for CG content, the MIV anchor is able to code the light field more efficiently. However, for CG content with transparencies in the scene, the depth maps can be misleading, as in the case of Cadillac and Fan.

With the exception of Hall, the MIV DSDE outperforms the MIV anchor. This is because, despite the best attempts for their estimation and refinement, the depth maps provided on the encoder side are not sufficiently accurate. As seen in the Kitchen sequence, the MIV DSDE anchor can even outperform MIV on CG content with high-quality depth maps. Similar to the MIV view, the performance of MIV DSDE is much better for perspective content than for ERP. Due to the higher resolutions of the omnidirectional sequences, there are fewer basic views available at the decoder side, which limits depth estimation. Also, the baselines between the views are wider as compared to the typical depth estimation scenario. Therefore, IVDE performs much better for perspective content. This is why, in the following chapters, we focus on DSDE methods for perspective content.

The MIV view and MIV DSDE outperform the MIV anchor in several sequences for low bitrates, *e.g.,* Carpark and Fencing. This reflects that the impact of strong depth compression leads to a higher degradation of synthesis performance, than the estimation of depth maps from compressed textures.

As there is no need for the intricate pruning and packing process, the TMIV encoder is considerably simpler for the MIV view and MIV DSDE than it is for the MIV anchor. Additionally, since full-frame atlases may be compressed more quickly than patch atlases, video encoding is faster.

Fig. 3.9 shows the synthesis results on a selected frame of a rendered target view, for matching bitrates and for all the anchors. In the first row, we show an example of the synthesis for the Museum sequence, which has a better coding performance for the MIV anchor. The synthesized views clearly confirm that the MIV view as well as MIV DSDE could not recover all the textures. In the case of the MIV view,

this is likely due to the pixel rate constraint. As Museum has a resolution of 2048 x 2048, the number of basic views that can be coded in the MIV view anchor is far too small, in order to reconstruct all 24 views with sufficient quality. In the case of the MIV DSDE anchor, the situation becomes more severe, as the number of transmitted views is too low to do high-performant depth estimation with IVDE. Consequently, a lot of occlusions remain visible in the final views. In the case of the MIV anchor, rendered target view has small cracks or patch-level compression block artifacts.

In general, the displayed views are sharper with the MIV DSDE anchor because the related depth maps are free of compression artifacts. On the other hand, if the coded subset of views is insufficient to carry out high-quality depth estimation with IVDE, occlusions may degrade the perceptual quality. The MIV view anchor is especially suitable for this kind of information because the pruning tools used in the MIV anchor have not been developed for non-Lambertian surfaces.

3.5 Conclusion

In this chapter, we provided a comprehensive review of the most recent iteration of the Test Model for MPEG Immersive Video (TMIV), which serves as the benchmark software for the MIV standard. As this thesis has been conducted since the initial stages of MIV standardization, we also outlined the distinctions between the common test conditions and TMIV versions used in our experiments. Additionally, we presented the testing conditions and analyzed the performance of different MIV anchors.

The following chapters will present the contributions achieved during the work on this thesis and give their detailed analysis.

(A) The MIV anchor.

(B) The MIV view anchor.

(C) The MIV DSDE anchor.

FIGURE 3.9: Comparison of synthesis results obtained by different anchors, Museum sequence.

Chapter 4

Performance of Video Codecs and Screen Content Tools for MIV

This chapter[1] focuses on the atlas compression as a potential improvement of an immersive video coding system. Namely, we evaluate the impact of different 2D video encoders and their configurations on the quality of the resulting target views in the immersive video setting. In essence, MIV is a pre-processing technology developed specifically for the compression of immersive video. It operates independently of the 2D video codecs used for the final compression of its atlases and aims to provide a more efficient and effective MVD coding process. The first part of this chapter provides a comprehensive understanding of the efficiency of MIV used with different state-of-the-art 2D codecs, while the second part focuses on the influence of different screen content coding tools in VVC on the performance of MIV.

4.1 Introduction

Plenty of video setups that require video compression have the ability to support multiple codecs [111, 112], thus making them "codec-agnostic." This means that they do not have a specific preference or dependence on a particular codec, and can work with a wide range of video compression technologies. This adaptability is a desirable characteristic as it allows the system to be flexible and increases its chances of industrial success. Furthermore, being codec-agnostic can also help future-proof the system, as it enables the integration of new codecs and advancements in video compression technology as they become available.

Moreover, different video coding standards have introduced the so-called screen content coding tools [113, 37, 114], carefully crafted to handle some computer screen characteristics (*e.g.* sharp edges, limited color palette, and recurring motifs), used for screen recordings, cloud gaming, or remote keyboard. Since MIV atlases contain a lot of repeated patterns, and additionally, the depth atlases contain a limited palette and texture-less regions, there has been some research on the influence of HEVC-SCC tools on MIV [115].

In this chapter, we focus on the MIV atlas coding efficiency. First, in Section 4.2, we recall the followed test conditions. Then, in Section 4.3, we provide a study of the performance of MIV, encompassing video codecs from all over the globe, which are previously detailed in Chapter 2. Finally, our study also shows a detailed analysis and performance of different VVC screen content coding tools and their influence on MIV atlas coding, in Section 4.4.

[1] The content of this chapter is based on the work we published in [27].

The results of these experiments are important for understanding the compatibility and competitiveness of MIV with various 2D codecs. They also provide valuable insights into how different encoding software and configurations can affect the quality of the rendered target views. These findings can help guide future research and development in the field of immersive video compression and can inform the selection of appropriate codecs and configurations for specific use cases.

4.2 Test conditions

In this chapter, we follow the CTC of MIV for the evaluation of different proposals, as detailed in Section 3.3. The test material for evaluation comprises both natural and computer-generated content captured using perspective and omnidirectional cameras. The material is categorized into mandatory and optional sequences within the CTC, where the optional sequences are more demanding but not mandatory for the evaluation process.

The test sequences are compressed using TMIV (version 10) and VVenC [95, 96], an implementation of VVC [97]. The VVenC is used in the "slow" preset with the encoder in the random access configuration [98]. The quantization parameters $\mathbf{QP_T}$ for the compression of texture videos are chosen as given in Section 3.3. The MIV metadata and the video bitstreams are decoded using TMIV decoder and VVdeC [99]. The target viewport from is rendered on the same positions as source cameras, to facilitate the objective quality evaluation. The objective evaluation is done by computing the Bjøntegaard delta (BD) rates [103, 104] in terms of Y-PSNR of synthesized views in comparison to the uncoded source views, as well as in terms of IV-PSNR [105].

4.3 MIV codec agnosticism

During the development of MIV, it was necessary to use HEVC or VVC as mandated by the CTC. However, there are now a multitude of coding standards available and no single codec is likely to dominate the market. Fortunately, MIV is designed to be compatible with any 2D codec and it can benefit from this competition. The purpose of this section is to verify the compatibility of MIV with the most recent codecs from major video coding organizations, including the VVC [97] from ISO/IEC MPEG and ITU-T VCEG, AV1 [39] from Alliance for Open Media [40], and AVS3 [41] from Audio Video Coding Standard Workgroup of China [42]. Our goal is to confirm that MIV atlases do not have any specific features that would negatively impact the efficiency of any 2D codec designed for traditional 2D content.

4.3.1 Software settings

In this study, we compare MIV with reference software alternatives, including VTM12 for VVC, HPM12 for AVS3, and AV1. The random access configuration as specified by each respective standardization body is used. No specific configuration adjustments were made to these codecs as our focus is not on comparing their competitiveness, but rather on showcasing the compatibility of MIV and overall performance. The main anchor of MIV, with VVenC using the "slow preset," is used as the reference point for comparison in all cases.

4.3.2 Results and discussion

Fig. 4.1 shows the decoded texture atlas quality, which is ultimately used for view synthesis. The coded texture and depth atlases are shown in Fig. 4.2. The associated encoder runtimes are shown in Tab. 4.1. In all the cases, MIV decoding times are very similar, and we do not show them here. For completeness, we provide the results of these experiments in terms of BD-rate in Tab. 4.2. However, this table should serve as a reference and not as a tool to compare these codecs with each other. Instead, we analyze each codec independently by comparing the performance per sequence with the average.

AV1 shows quite a good performance with Painter, Carpark, and Hall sequences. It performs the worst for Group, Kitchen, and ChessPieces. Overall, stronger fluctuations across the sequences are noticeable. HPM12 performs rather stable across all sequences and shows most difficulties with the Fan and Group sequences. Similar to AV1, it performs well for Painter and Hall. VTM12 performs better for Group, ChessPieces, and Frog, instead of Painter and Hall. Similar to HMP12, its performance is rather stable across the sequences. Based on these findings, it appears that none of the codecs conflicts with the characteristics of MIV atlases. Thus, the capability of the MIV standard to be compatible with any codec is confirmed in practice.

FIGURE 4.1: Details from the compressed texture atlases of the Painter sequence, matching bitrate, all tested encoders. The first row shows the uncompressed source texture. Second row: cropped source, third row: VVenC, fourth row: VTM12, fifth row: AV1, last row: HPM12. The second and fourth columns represent the difference between the encoded atlas and the source.

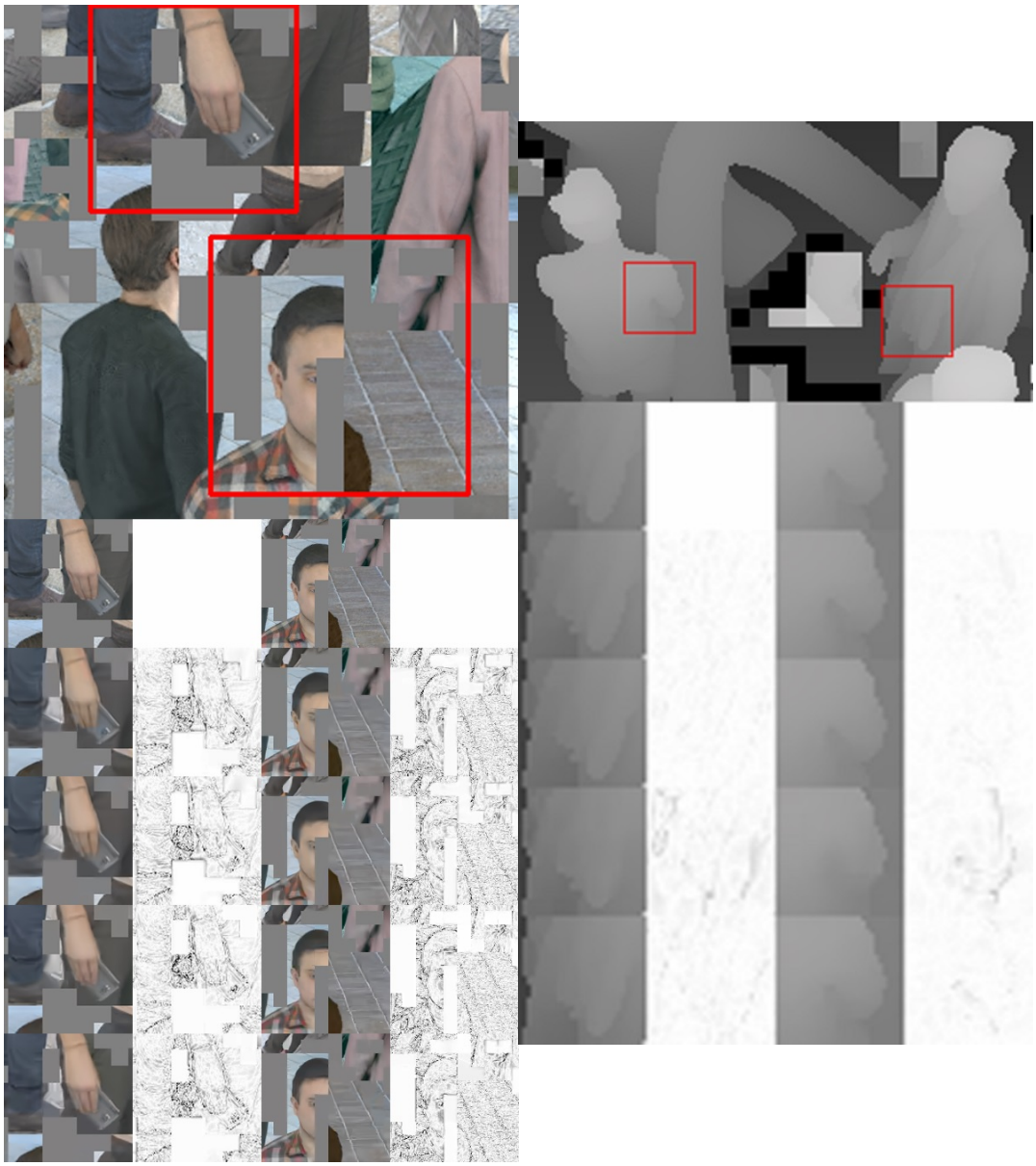


FIGURE 4.2: Details from the compressed atlases of the Group sequence, matching bitrate, all tested encoders: a) texture patch atlas, b) depth patch atlas. The first row shows the uncompressed source. Second row: cropped source, third row: VVenC, fourth row: VTM12, fifth row: AV1, last row: HPM12. The second and fourth columns represent cropped images of the difference between the encoded atlas and the source.

TABLE 4.1: Video encoding runtimes for different codecs, compared to the A17 anchor (using VVenC slow preset).

Mandatory content			
	Video encoding [%]		
Sequence	VTM12	AV1	HPM12
Painter	719.0	131.6	1859.7
Frog	713.4	163.0	2117.2
Carpark	666.7	149.5	1892.0
ClassroomVideo	647.6	171.5	2342.3
Museum	622.6	250.0	1694.3
Fan	659.5	209.2	2241.5
Kitchen	543.9	83.6	1285.3
Chess	666.9	181.6	1479.8
Group	538.7	131.3	1625.7
Average	**642.0**	**163.5**	**1837.5**
Optional content			
Fencing	779.2	143.8	1580.2
Hall	715.9	82.5	1252.5
Street	637.8	130.7	1736.8
Mirror	607.8	142.9	1790.2
ChessPieces	714.0	153.8	1407.3
Hijack	665.8	133.3	1846.7
Cadillac	717.3	152.8	2069.9
Average	**691.1**	**134.3**	**1669.1**

TABLE 4.2: MIV objective performance with VTM12, AV1 and HPM12, compared to the A17 anchor (using VVenC slow preset).

Mandatory content												
	High BD-Rate PSNR [%]			Low BD-Rate PSNR [%]			High BD-Rate IV-PSNR [%]			Low BD-Rate IV-PSNR [%]		
Sequence	VTM12	AV1	HPM12	VTM12	AV1	HPM12	VTM12	AV1	HPM12	VTM12	AV1	HPM12
Painter (NC)	-2.0	-35.2	-14.3	-2.3	-45.3	-19.0	5.7	-34.0	-9.8	3.9	-44.0	-14.6
Frog (NC)	-6.9	8.9	6.4	-6.6	-1.8	3.4	7.8	7.8	11.2	5.2	0.1	7.2
Carpark (NC)	-3.6	-11.6	0.5	-3.7	-21.1	-4.9	6.0	-16.7	0.7	4.0	-24.1	-4.9
Class.Video (CG)	-1.2	18.4	17.2	1.4	1.8	8.3	7.2	10.0	17.4	7.9	2.6	11.4
Museum (CG)	-2.4	45.1	9.7	-2.8	33.0	5.9	2.5	14.3	5.7	2.5	13.6	5.0
Fan (CG)	-4.3	41.5	60.7	-3.7	19.4	44.4	-1.1	17.7	50.8	0.9	-1.8	37.1
Kitchen (CG)	-4.7	100.1	17.0	-5.5	71.6	16.9	1.7	31.1	10.1	4.1	24.2	6.7
Chess (CG)	-5.6	70.9	21.3	-7.3	28.8	10.5	0.5	10.2	17.1	-0.4	3.6	6.2
Group (CG)	-14.0	155.3	47.6	-11.6	86.4	34.2	-3.0	65.5	28.4	-1.4	42.4	19.5
Average	**-5.0**	**43.7**	**18.4**	**-4.7**	**19.2**	**11.1**	**3.0**	**11.8**	**14.6**	**3.0**	**1.8**	**8.2**
Optional content												
Fencing (NC)	-5.7	-4.3	-3.8	-5.6	-18.6	-9.9	2.1	2.0	-0.5	1.2	-14.2	-8.4
Hall (NC)	-5.6	-67.1	-37.3	-6.4	-68.3	-37.2	4.5	-64.5	-26.1	5.4	-66.2	-28.2
Street (NC)	-1.9	5.4	4.2	-1.8	-11.2	-6.1	6.6	-6.8	2.4	5.9	-14.6	-5.3
Mirror (NC)	-2.7	4.7	9.8	-3.5	-2.6	7.5	3.7	-1.8	11.1	3.1	-5.9	8.9
ChessPieces (CG)	-6.6	61.9	14.0	-7.1	26.1	6.1	0.8	9.6	16.5	0.0	5.1	5.8
Hijack (CG)	-4.6	25.0	13.5	-5.5	3.8	7.0	-1.7	-6.0	11.8	-1.5	-11.8	3.0
Cadillac (CG)	-4.5	49.1	20.0	-5.6	32.9	16.9	2.3	11.0	7.8	0.7	3.9	4.8
Average	**-4.5**	**10.7**	**2.9**	**-5.1**	**-5.4**	**-2.2**	**2.6**	**-8.1**	**3.3**	**2.1**	**-14.8**	**-2.8**

4.4 Atlas coding using SCC tools in VVC

HEVC has been the 2D codec for the evaluation of TMIV throughout most of the development stages of MIV. After VVC was finalized, the MPEG experts decided to use VVC instead of HEVC [116]. However, no significant effort to tune the encoder or to take advantage of the versatility aspect of VVC has been done. Especially for depth coding, VVC comes with interesting edge-preserving tools like geometrical

partitioning. Screen content coding (SCC) tools have also shown the potential to compress MVD content efficiently with HEVC [113]. In this section, we investigate the performance of intra block copy (IBC), block-based differential pulse-code modulation (BDPCM), and palette mode (PLT) on different types of atlases:

- Basic-view (BV) atlases, containing basic views and a very small amount of patches from additional views.
- Additional-view (AV) patch atlases, containing only small patches from additional views.
- Texture atlases, which consist of only texture component, both for basic view and additional view atlases.
- Depth atlases, which consist of only depth component, both for basic view and additional view atlases.

4.4.1 Screen content coding tools

This section gives an overview of tested screen content coding tools, explaining their methodology and implementation in the VVC standard.

IBC searches for the intra-picture similarity on a block basis, where the prediction of the current coding block is done by the reconstructed reference block from the same picture. It has a great performance in screen content coding, in terms of BD-rate, at a cost of increased complexity [37]. The search range is restricted to the already coded coding tree unit (CTU) to the left of the current CTU, which is a strong limitation for atlas coding, as otherwise, IBC may reach a similar performance as the multiview extension of HEVC (MV-HEVC). However, here we do not perform any normative changes to the VVC codec.

BDPCM is a prediction technique that is grounded on the residual DPCM [117, 38]. It serves as an alternative intra prediction mode, which is better suitable to avoid errors in predictions of long pixel distances. In VVC, BDPCM is limited to the same block sizes as the transform skip mode, and no boundary filtering is applied between the neighboring blocks predicted with this method.

Palette mode has proven to be beneficial for CG content with simple graphics. It encompasses the coding of the palette (different samples) and coding of the spatial position index of its coding unit. This technique is considered a special coding mode in VVC, together with the intra-prediction, inter-prediction, and IBC modes. Palette mode is deemed to be less efficient and less complex than IBC [37].

4.4.2 Results and discussion

A summary of the performance of MIV with VTM-SCC tools is shown in Tab. 4.3. In all the cases, we compare the alternatives with the TMIV+VTM anchor, to isolate the impact of the screen content coding tools. Every tool provides additional benefit to the overall coding gain. IBC performs very well on Fan, ClassroomVideo, and ChessPieces. BDPCM provides a small, but consistent gain over all sequences. Taking into account the video encoding runtime, shown in Tab. 4.4, the benefit of BDPCM comes with only a minor complexity increase over IBC alone. Consequently, we will show these tools jointly in the following sections to reduce the amount of data to present. However, the additional coding gain of PLT for some sequences comes with added complexity. For a few sequences, PLT reduces the performance. Fan is an interesting exception, as the coding gain is significant. In order to further analyze if these

tools are particularly beneficial for certain atlas types, we need to look at each type of content independently.

TABLE 4.3: TMIV+VTM objective performance with IBC, IBC+BDPCM, and IBC+BDPCM+PLT, as compared to the TMIV+VTM anchor.

Mandatory content - Objective Performance												
	High BD-Rate PSNR [%]			Low BD-Rate PSNR [%]			High BD-Rate IV-PSNR [%]			Low BD-Rate IV-PSNR [%]		
Sequence	IBC	IBC BDPCM	IBC BDPCM PLT	IBC	IBC BDPCM	IBC BDPCM PLT	IBC	IBC BDPCM	IBC BDPCM PLT	IBC	IBC BDPCM	IBC BDPCM PLT
Painter (NC)	-0.9	-2.1	-3.3	-1.0	-1.9	-3.1	-0.3	-2.1	1.0	-0.4	-1.5	1.3
Frog (NC)	-0.6	-1.1	-2.3	-0.6	-1.1	-2.5	-0.7	-0.8	2.3	-0.9	-0.8	2.2
Carpark (NC)	-1.3	-2.2	-3.2	-1.4	-2.0	-3.1	-1.9	-2.6	0.0	-1.8	-2.2	0.6
Class.Video (CG)	-6.3	-8.8	-10.4	-6.8	-9.9	-11.9	-5.9	-7.6	-4.1	-6.8	-9.3	-7.4
Museum (CG)	-1.9	-2.7	-3.5	-2.6	-3.2	-4.0	-1.7	-2.3	2.2	-1.9	-2.7	1.4
Fan (CG)	-7.2	-9.3	-26.3	-7.0	-8.9	-21.7	-6.3	-7.8	-21.3	-6.3	-7.8	-16.6
Kitchen (CG)	-2.1	-3.8	-3.4	-2.8	-3.9	-5.6	-2.3	-3.4	1.2	-2.9	-4.2	0.1
Chess (CG)	-5.0	-5.2	-7.6	-4.6	-5.8	-8.5	-2.7	-4.0	4.0	-3.3	-4.9	1.1
Group (CG)	-2.0	-3.9	-7.3	-2.9	-4.5	-8.3	-1.7	-2.7	1.3	-2.1	-3.4	1.0
Average	**-3.0**	**-4.3**	**-7.5**	**-3.3**	**-4.6**	**-7.6**	**-2.6**	**-3.7**	**-1.5**	**-3.0**	**-4.1**	**-1.8**
Optional content - Objective Performance												
Fencing (NC)	-1.2	-0.9	-2.2	-1.1	-1.5	-2.2	-1.2	-1.0	1.8	-1.1	-1.4	2.0
Hall (NC)	0.1	-1.5	-0.7	-0.7	-1.6	-1.4	0.0	-1.3	1.4	-0.4	-1.3	2.4
Street (NC)	-0.2	-0.5	-0.7	-0.6	-0.9	-1.3	-1.7	-1.2	2.4	-1.6	-1.4	3.0
Mirror (NC)	-1.6	-2.2	-3.9	-1.6	-2.3	-4.0	-1.3	-1.7	-1.3	-1.4	-2.0	-1.8
ChessPieces (CG)	-6.5	-7.4	-10.0	-5.6	-6.9	-10.5	-4.2	-5.7	1.4	-4.7	-6.4	-2.0
Hijack (CG)	-2.6	-4.2	-6.2	-2.8	-4.5	-6.6	-2.5	-4.0	-3.8	-2.8	-4.4	-4.4
Cadillac (CG)	-2.7	-3.7	-2.7	-2.8	-4.5	-2.8	-2.4	-3.6	-2.4	-2.6	-4.2	-2.6
Average	**-2.1**	**-2.9**	**-3.8**	**-2.2**	**-3.2**	**-4.1**	**-1.9**	**-2.6**	**-0.1**	**-2.1**	**-3.0**	**-0.5**

TABLE 4.4: Video encoding runtimes for different SCC tools, compared to the TMIV+VTM.

Mandatory content			
	Video encoding [%]		
Sequence	IBC	IBC BDPCM	IBC BDPCM PLT
Painter	115.1	116.6	126.1
Frog	114.5	115.1	125.8
Carpark	115.4	116.4	129.9
ClassroomVideo	112.3	113.5	126.4
Museum	118.3	119.4	141.3
Fan	119.0	123.6	134.9
Kitchen	122.2	122.3	146.1
Chess	116.5	117.6	135.5
Group	123.0	124.1	145.8
Average	**117.4**	**118.8**	**134.6**
Optional content			
Fencing	111.2	112.2	120.4
Hall	112.2	112.8	122.7
Street	115.3	116.0	131.0
Mirror	116.9	118.1	133.7
ChessPieces	112.2	112.8	129.4
Hijack	114.7	114.7	126.6
Cadillac	112.3	112.0	112.3
Average	**113.5**	**114.1**	**125.1**

TABLE 4.5: IBC+BDPCM texture atlas results.

Mandatory content								
	High BD-Rate PSNR [%]		Low BD-Rate PSNR [%]		High BD-Rate IV-PSNR [%]		Low BD-Rate IV-PSNR [%]	
Sequence	BV Atlas	AV Atlas	BV Atlas	AV Atlas	BV Atlas	AV Atlas	BV Atlas	AV Atlas
Painter (NC)	0.1	-0.5	0.2	-0.7	0.2	-0.4	0.6	-0.3
Frog (NC)	-0.2	-0.1	-0.2	-0.2	0.1	0.2	0.1	0.2
Carpark (NC)	-2.4	-0.3	-2.3	-0.5	-2.9	-0.9	-2.6	-0.7
Class.Video (CG)	-1.0	-1.2	-0.6	-1.0	-0.4	-0.8	-0.2	-0.8
Museum (CG)	-1.4	-1.0	-1.4	-1.1	-0.9	-0.6	-0.7	-0.5
Fan (CG)	-6.8	-5.8	-5.6	-5.1	-4.0	-3.1	-3.2	-2.9
Kitchen (CG)	-0.8	-1.8	-0.1	-1.1	-0.4	-1.4	-0.5	-1.4
Chess (CG)	-1.9	-2.6	-1.5	-2.5	-0.5	-1.3	-0.5	-1.5
Group (CG)	-2.4	-2.6	-2.5	-2.4	-0.9	-1.3	-1.2	-1.4
Average	**-1.9**	**-1.8**	**-1.6**	**-1.6**	**-1.1**	**-1.1**	**-0.9**	**-1.0**
Optional content								
Fencing (NC)	0.6	0.3	-0.1	-0.4	0.4	0.1	-0.1	-0.3
Hall (NC)	-0.8	-1.3	-1.3	-1.3	-0.4	-1.0	-0.9	-1.1
Street (NC)	-0.5	0.5	-0.9	-0.2	-1.2	-0.1	-1.4	-0.6
Mirror (NC)	-0.8	-0.9	-0.7	-0.9	-0.3	-0.4	-0.3	-0.5
ChessPieces (CG)	-3.1	-3.9	-1.5	-2.7	-0.8	-1.9	-0.9	-2.3
Hijack (CG)	-1.1	-1.7	-0.9	-2.1	-0.7	-1.4	-0.8	-1.9
Cadillac (CG)	-1.1	-0.8	-1.9	-1.7	-1.0	-0.8	-1.4	-1.2
Average	**-1.0**	**-1.1**	**-1.0**	**-1.3**	**-0.6**	**-0.8**	**-0.8**	**-1.1**

Texture atlas coding

The SCC results for the texture atlases alone are shown in Tab. 4.5 for IBC and BDPCM as well as in Tab. 4.6 using additionally PLT. We do not observe a significant difference between BV and AV atlases in that case, indicating that there is no particular advantage over different texture atlas types. For BV atlases, IBC is not able to efficiently remove a significant amount of redundancies due to limitations of its implementation in the standard. Due to the strong differences between the patches in the AV atlases, the benefit there is also limited. Nevertheless, PLT proves particularly useful for AV atlas coding. The benefit increases with the amount of repetitive textures in the scene as it is the case for Fan, Chess and ChessPieces. However, following the CTC of MIV, the chroma component is not taken into account in our objective metrics. The Fan sequence shown in Fig. 4.3 indicates that the SCC tools may negatively impact the color component. While some structures may be more efficiently recovered using SCC tools, we do not see a general benefit of SCC tools for the usage of texture atlases.

Depth atlas coding

The SCC results related to depth atlas coding are shown in Tab. 4.7 for IBC and BDPCM and in Tab. 4.8 for PLT additionally. In the case of BV atlases, similar constraints for IBC are given as for texture atlases. However, it is much simpler to find matching structures in the depth domain than in the texture domain. Furthermore, the performance of SCC tools is better for depth AV atlases. While variations of colors make the coding of textures more challenging, this is not the case for depth maps. Consequently, AV, as well as BV atlases, can be efficiently coded using SCC tools. In comparison to the performance for texture atlases, we see a huge benefit of utilizing the SCC tools for the coding of depth atlases. The same holds for PLT as significant coding gain is observed over IBC+BDPCM. An example is shown in Fig. 4.3, which shows that structures of the depth are much better preserved using

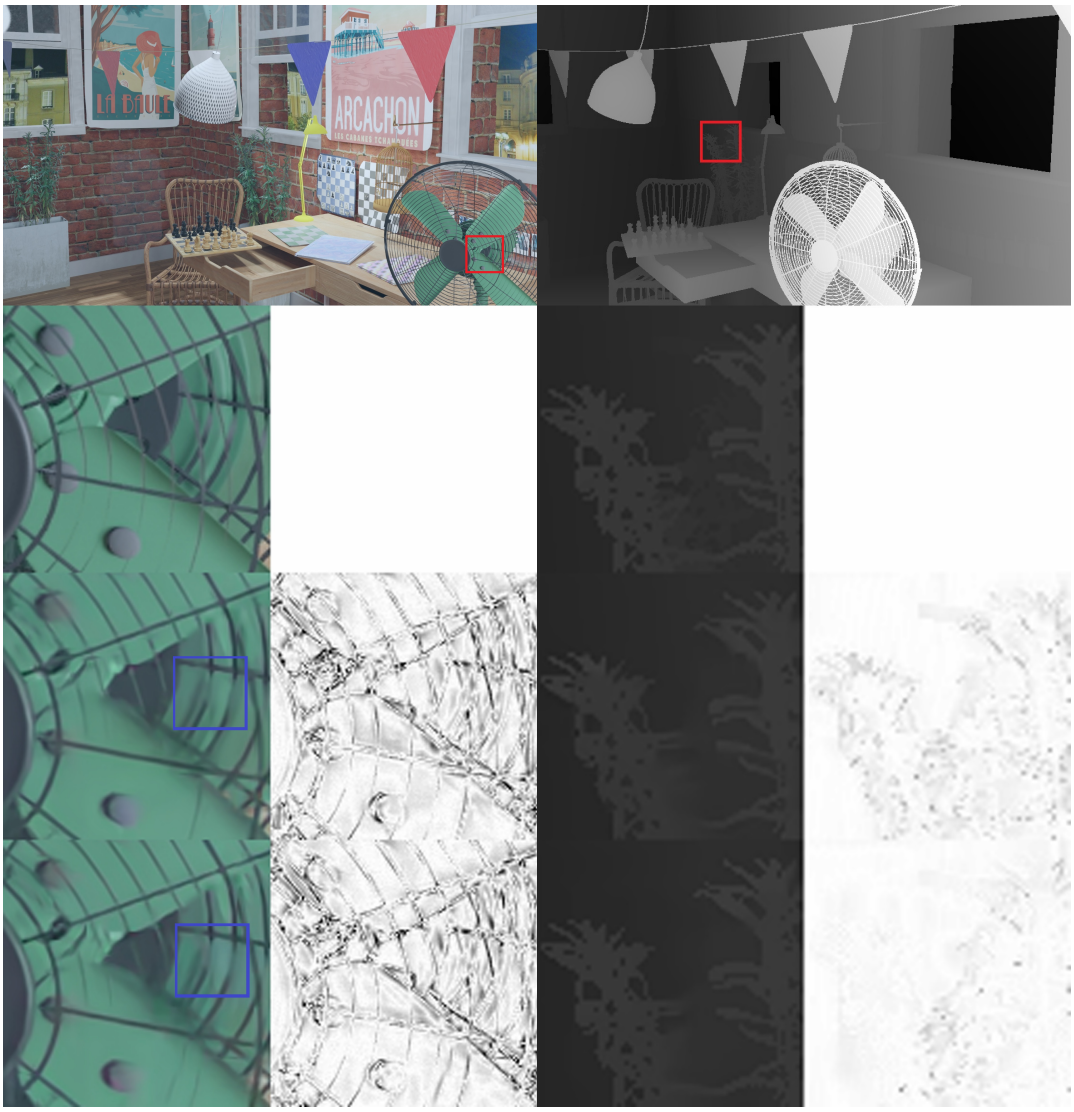


FIGURE 4.3: Details from the compressed atlases of the Fan sequence, matching bitrate: a) texture patch atlas, b) depth patch atlas. First row: source view, second row: cropped source, third row: encoded with VTM, fourth row: encoded with VTM and SCC tools. The blue block highlights an example of color change if SCC tools are used.

SCC tools. Consequently, we see a general benefit of SCC tools regarding depth atlas coding. In case of concerns related to complexity, these tools should be prioritized for the coding of depth atlases instead of texture atlases.

TABLE 4.6: IBC+BDPCM+PLT texture atlas results.

Mandatory content								
	High BD-Rate PSNR [%]		Low BD-Rate PSNR [%]		High BD-Rate IV-PSNR [%]		Low BD-Rate~ IV-PSNR [%]	
Sequence	BV Atlas	AV Atlas	BV Atlas	AV Atlas	BV Atlas	AV Atlas	BV Atlas	AV Atlas
Painter (NC)	0.1	-1.2	0.2	-1.3	5.8	4.0	5.5	3.3
Frog (NC)	-0.3	-0.5	-0.3	-0.6	5.7	5.1	6.0	5.3
Carpark (NC)	-2.7	-0.5	-2.9	-0.8	1.8	3.9	2.7	4.2
Class.Video (CG)	-0.3	-1.6	-0.0	-1.1	7.6	3.1	6.3	2.9
Museum (CG)	-1.7	-1.7	-1.7	-1.5	5.3	4.3	5.5	4.3
Fan (CG)	-27.9	-26.2	-20.9	-20.2	-18.2	-16.8	-10.7	-10.1
Kitchen (CG)	0.9	-1.4	-0.6	-2.5	6.5	2.9	6.9	2.7
Chess (CG)	-3.7	-4.7	-3.0	-4.5	11.3	8.9	10.6	7.6
Group (CG)	-6.2	-6.0	-6.9	-6.3	4.9	3.2	6.5	4.1
Average	**-4.6**	**-4.9**	**-4.0**	**-4.3**	**3.4**	**2.1**	**4.4**	**2.7**
Optional content								
Fencing (NC)	-0.5	-0.6	-0.1	-0.7	5.0	4.5	5.8	5.0
Hall (NC)	0.3	-0.2	-0.9	-1.6	2.9	2.5	3.0	1.9
Street (NC)	-0.5	1.9	-1.1	0.1	3.1	4.8	3.9	4.3
Mirror (NC)	-1.3	-1.8	-1.0	-1.4	2.2	1.5	2.3	1.7
ChessPieces (CG)	-4.0	-6.0	-3.4	-6.0	11.8	7.8	8.5	5.1
Hijack (CG)	-1.4	-2.7	-1.1	-3.0	2.5	0.8	2.4	0.7
Cadillac (CG)	-1.4	-0.6	-1.4	-0.4	-1.0	-0.2	-1.0	-0.1
Average	**-1.3**	**-1.4**	**-1.3**	**-1.8**	**3.8**	**3.1**	**3.6**	**2.6**

TABLE 4.7: IBC+BDPCM depth atlas results.

Mandatory content								
	High BD-Rate PSNR [%]		Low BD-Rate PSNR [%]		High BD-Rate IV-PSNR [%]		Low BD-Rate IV-PSNR [%]	
Sequence	BV Atlas	AV Atlas	BV Atlas	AV Atlas	BV Atlas	AV Atlas	BV Atlas	AV Atlas
Painter (NC)	-2.8	-3.1	-2.2	-2.5	-2.8	-3.2	-1.8	-2.2
Frog (NC)	-1.9	-2.5	-1.7	-2.3	-1.7	-2.4	-1.5	-2.2
Carpark (NC)	-2.3	-2.9	-1.8	-2.4	-2.6	-3.2	-1.9	-2.6
Class.Video (CG)	-15.7	-19.0	-12.4	-16.7	-15.8	-19.2	-13.2	-17.2
Museum (CG)	-4.9	-7.6	-4.0	-6.7	-4.9	-7.7	-3.8	-6.6
Fan (CG)	-10.9	-9.6	-10.2	-8.8	-10.1	-8.9	-9.4	-8.0
Kitchen (CG)	-8.7	-12.1	-7.5	-11.4	-8.8	-12.0	-7.6	-11.5
Chess (CG)	-8.9	-11.9	-7.7	-10.9	-8.6	-11.6	-7.6	-10.8
Group (CG)	-7.5	-8.5	-6.2	-7.7	-7.2	-8.2	-6.2	-7.6
Average	**-7.1**	**-8.6**	**-6.0**	**-7.7**	**-6.9**	**-8.5**	**-5.9**	**-7.6**
Optional content								
Fencing (NC)	-1.8	-2.5	-1.9	-2.5	-1.9	-2.6	-1.9	-2.5
Hall (NC)	-1.4	-2.2	-1.4	-2.3	-1.3	-2.0	-1.1	-2.0
Street (NC)	-1.3	-1.3	-1.4	-1.6	-1.8	-1.7	-1.7	-1.8
Mirror (NC)	-3.3	-4.2	-2.9	-3.8	-3.0	-4.1	-2.8	-3.7
ChessPieces (CG)	-8.3	-12.6	-6.2	-10.7	-7.9	-12.2	-6.4	-10.9
Hijack (CG)	-5.6	-7.4	-4.5	-6.3	-5.5	-7.3	-4.5	-6.3
Cadillac (CG)	-5.9	-6.5	-5.9	-6.3	-5.9	-6.5	-5.6	-6.1
Average	**-3.9**	**-5.2**	**-3.5**	**-4.8**	**-3.9**	**-5.2**	**-3.4**	**-4.8**

TABLE 4.8: IBC+BDPCM+PLT depth atlas results.

Sequence	High BD-Rate PSNR [%]		Low BD-Rate PSNR [%]		High BD-Rate IV-PSNR [%]		Low BD-Rate~ IV-PSNR [%]	
	BV Atlas	AV Atlas	BV Atlas	AV Atlas	BV Atlas	AV Atlas	BV Atlas	AV Atlas
Mandatory content								
Painter (NC)	-2.9	-5.3	-2.1	-4.8	0.7	-2.5	2.0	-1.6
Frog (NC)	-3.7	-5.7	-3.3	-5.6	-0.5	-3.4	-0.1	-3.1
Carpark (NC)	-3.9	-5.5	-3.2	-4.7	-2.0	-3.9	-0.8	-2.6
Class.Video (CG)	-17.3	-24.7	-14.1	-21.6	-15.6	-24.3	-13.2	-21.5
Museum (CG)	-5.6	-9.8	-4.4	-8.8	-4.4	-8.9	-3.0	-7.7
Fan (CG)	-23.2	-27.0	-19.6	-22.8	-20.2	-24.7	-16.0	-19.9
Kitchen (CG)	-8.9	-15.7	-8.4	-15.6	-7.5	-14.3	-6.0	-14.0
Chess (CG)	-11.2	-15.4	-10.1	-14.6	-7.7	-12.3	-7.1	-11.8
Group (CG)	-9.1	-11.2	-8.3	-10.6	-6.9	-9.0	-5.9	-8.2
Average	**-9.5**	**-13.4**	**-8.2**	**-12.1**	**-7.1**	**-11.5**	**-5.6**	**-10.0**
Optional content								
Fencing (NC)	-2.7	-4.2	-2.5	-4.0	-0.3	-1.9	0.4	-1.4
Hall (NC)	-0.6	-2.0	-1.0	-2.6	1.3	-0.2	2.7	0.6
Street (NC)	-1.4	-2.4	-1.7	-2.8	0.6	-0.8	1.6	-0.4
Mirror (NC)	-5.5	-7.7	-4.7	-7.0	-4.3	-6.9	-3.7	-6.1
ChessPieces (CG)	-10.8	-16.3	-9.0	-14.9	-6.7	-12.5	-5.8	-11.9
Hijack (CG)	-7.3	-11.1	-5.9	-9.5	-6.6	-10.4	-5.2	-8.9
Cadillac (CG)	-4.4	-4.3	-4.0	-4.1	-4.3	-4.2	-3.8	-4.0
Average	**-4.7**	**-6.9**	**-4.1**	**-6.4**	**-2.9**	**-5.3**	**-2.0**	**-4.6**

4.5 Conclusion

MIV produces atlases with the necessary data to properly render any viewpoint on the client side. Advantageously, the standard allows encoding these atlases with any existing 2D video codec. The results of encoding MIV atlases with VVC, AV1, and AVS3 are reported, and the MIV agnosticism to the 2D codec is confirmed through experimental results. The study also investigated the impact of three coding tools designed for screen content coding on the encoding of MIV atlases with VVC. These tools, namely intra block copy, BDPCM, and palette mode, are evaluated for their individual impact on different components of MIV atlases, such as texture and depth, and basic or additional views. This provides valuable insights into the optimization of encoders for practical MIV implementations. Overall, the results of this study demonstrate the strength and versatility of MIV, making it an attractive choice for multiview video coding applications.

Chapter 5

Performance of Depth Estimation Techniques for MIV

This chapter[1] focuses on the depth estimation process as a potential improvement of an immersive video coding system. It outlines the study on depth estimation tools in the scope of the MIV framework, which is the result of collaborative work at Orange Labs. Preliminary results with different classical and learning-based depth estimators for the use case of immersive video transmission were previously demonstrated in my internship report [118]. That work, together with the work on the DSDE system [25], paved the way for the following studies on the performance of various depth estimation techniques in the MIV DSDE framework, as well as the comparative study of conventional and learned depth estimation techniques focused solely on the quality of MIV rendering, with an expanded selection of depth estimators. The first study will be detailed in this chapter. A summary of the second study can be found in Appendix A of this thesis because it is led by another PhD student.

5.1 Introduction

View synthesis and depth estimation are essential non-normative stages in the MIV processing pipeline. Here we focus on the depth estimation step, while the view synthesis is addressed in Chapter 7. The depth estimators are generally designed to perform well for a specific use case, as detailed in Section 2.4. When it comes to the volumetric video transmission system, the depth estimator requirements become more challenging to fulfill due to the nature of its content. First of all, the depth estimator should be able to accommodate numerous input views (since the light field is sampled by multiple cameras) and also provide inter-view consistency among different depth maps [64]. Also, high resolutions and frame rates are required to maintain a sense of immersion for the user [119]. The depth estimator needs also to be flexible towards different projection formats (equirectangular, perspective, and orthographic) and camera arrangements with different sparsities and setups (linear, two-dimensional matrix, spherical outward looking). Finally, the domain shift issue can significantly degrade the quality of the produced depth map [54]. As a result, the depth estimator must model both the characteristics of computer-generated and natural content, which is difficult because it necessitates a massive number of ground-truth samples.

On the other hand, it has been shown that common metrics used to evaluate the image and video quality, such as PSNR and SSIM, have a weak correlation with multiple stereo matching (SM) metrics, such as *bad 2.0*, *avgerr*, *rms*, and *a95* [72, 120]. These SM metrics are evaluating how close the depth estimation approaches are to

[1] The content of this chapter is mostly based on the work we published in [26].

the ground truth and are commonly used to train and evaluate learning-based depth estimators. Moreover, the same study has shown that some rendering approaches can produce better results if using estimated depth maps instead of ground-truth ones. According to the study [120], if a quality metric relies on an interpolation algorithm with weaknesses, this may result in the wrong quality assessment (in *rms* errors).

To enable a variety of applications, the MIV DSDE profile was developed to be compatible with a wide range of depth estimators, as further detailed in Section 5.2.1. The current MIV DSDE profile employs IVDE as a reference tool for estimating depth maps at the decoder side, whereas this chapter conducts a study of carefully chosen depth estimators to cover both the most recent data-driven deep learning approaches as well as IVDE and another conventional method. Since depth map estimation is just a step in the processing pipeline of an immersive video system, not the end goal, this study examines the impact of different depth estimators on the compression and rendering performance of MIV in DSDE mode, which makes it the first depth estimation study to take compressed textures into account. Section 5.2.2 recalls the test conditions followed in our experiments, while Section 5.2.3 presents the results and discussion.

5.2 Depth estimation tools for the MIV DSDE system

This section gives an overview of the required characteristics of a depth estimator used at the decoder side in the context of immersive video. Furthermore, it details the test conditions followed in our experimental setting, and the results obtained with different depth estimation techniques, with a discussion. The goal of our experiments is to investigate the potential benefits of using data-driven depth estimation techniques and to evaluate the impact of different depth estimators on the compression and rendering performance of MIV in DSDE mode.

5.2.1 Decoder-side depth estimator

The requirements listed here should be addressed by depth estimation methods used at the decoder side in the context of immersive video. Firstly, in the DSDE mode, the assumption of having an uncompressed, fully sampled light field is not met. In the DSDE context, a depth estimator has decoded (therefore, compressed) views as input and has to deal with possibly severe distortions (edge or color shift in the image, blurring). This can impair its inter-view matching capabilities. Moreover, the number of available views used for depth estimation in DSDE mode may be quite scarce, which may result in difficulties finding the matches for stereo or multi-view methods, thus creating a possible need for monocular depth estimation. Secondly, the DSDE system needs to conform to very strict complexity constraints in order to enable high resolutions and high frame rates, which may require some compromises in terms of performance. Finally, as will be shown in Chapter 6, a decoder-side depth estimator may have to handle small patch areas that contain less information, which may make the depth estimation process more difficult. The current MIV DSDE system, as described in Chapter 2, is only used for fully transmitted (non-pruned) frames.

5.2.2 Test conditions

In this study, we follow the test conditions defined by the MPEG CTC document [121], as presented in Section 3.3, to ensure impartial testing among various experimental designs. The TMIV reference software [122] (version 9) processes input textures in the DSDE mode to produce atlases and corresponding metadata. Then, the atlases are encoded and decoded with VVenC and VVdeC [95], following five different rate points (RP), as given by the CTC. Then, the depth estimation process is initiated before the view synthesis. The TMIV renderer renders the target views in the same positions as the input views, to enable objective quality assessment. The multi-view test video sequences used for our evaluation are of high resolution, perspective, and rectified. The test dataset contains NC and CG content. The performance is assessed through the quality of rendered views, with the following metrics: Y-PSNR, IV-PSNR [105], which is designed to consider rendering artifacts. Additionally, to give a broad comparison that accounts for the various types of distortions, the following machine-learning-based quality metrics are used: LPIPS [110] and VMAF [109].

We test four depth estimators overall, starting with Immersive Video Depth Estimation (IVDE) [56] and Depth Estimation Reference Software (DERS) [55], which are the current and previous MPEG reference software for depth estimation, respectively. Next to these two classical approaches, we test two end-to-end stereo matching learning-based methods, the Guided Aggregation Network (GA-Net) [58], and the Group-wise Correlation Stereo Network (GwcNet) [57]. Chapter 2 provides an overview of all tested depth estimators. IVDE and DERS do not place any limitations on camera structures, making them appropriate for immersive video settings. Both IVDE and DERS propose a similar strategy for estimating depth—the optimization of an energy function based on the graph structure—but they differ greatly on many technical levels. For example, IVDE uses superpixels as the estimate unit, whereas DERS uses pixels. GA-Net and GwcNet are end-to-end stereo matching networks based on cost volume matching with 3D convolution. These techniques were selected as one of the highly recognized techniques in the literature. For our evaluation of the performance of both techniques in the MIV environment, we have used the model pre-trained on the KITTI dataset [69]. The disparity maps produced by GA-Net and GwcNet are subsequently converted into depth maps [61].

5.2.3 Results and discussion

Table 5.1 and Table 5.2 present the rendering performance of all tested depth estimators for five different RPs. The depth estimator employed at the decoder side is the only difference in the framework, hence the total bitrates for the several tested approaches in this section are equivalent. As a result, it is able to compare the rendered views' quality directly, narrowing the focus to the variations in quality brought on by various depth estimating techniques. The best average results in the objective comparison between PSNR and IV-PSNR can be shown for DERS, but the results vary for individual sequences since, for almost half of the sequences, either DERS or IVDE has the best quality. DSDE has a better pixel-to-pixel correlation between the synthesized view and the input view, which results in higher quality in PSNR-based methods. The average results suggest that IVDE, which was tailored to manage such compression-induced artifacts, provides the highest quality for low bitrates when compared to VMAF and LPIPS, which were demonstrated to have an excellent correlation with perceived subjective quality. Even though the objective results show that synthesis results based on IVDE and DERS depth maps are better on average

TABLE 5.1: The comparison of average quality of rendered views for different depth estimators used in MIV DSDE, in Y-PSNR and IV-PSNR metrics.

Sequence	Rate	Y-PSNR [dB] (↑)				IV-PSNR [dB] (↑)			
		IVDE	DERS	GA-Net	GwcNet	IVDE	DERS	GA-Net	GwcNet
Fan	RP1	32.48	**32.82**	29.42	29.58	40.51	**40.90**	36.96	37.12
	RP2	32.30	**32.66**	29.35	29.49	40.32	**40.66**	36.86	37.02
	RP3	31.75	**32.11**	29.05	29.13	39.62	**40.02**	36.43	36.58
	RP4	30.37	**30.51**	28.18	28.15	37.91	**38.00**	35.25	35.33
	RP5	**28.22**	28.06	26.56	26.02	**35.18**	35.05	33.07	32.26
Kitchen	RP1	**35.52**	35.26	31.48	29.18	**43.78**	41.70	39.12	36.57
	RP2	**34.93**	33.95	31.25	29.04	**43.03**	40.36	38.83	36.36
	RP3	**34.13**	33.18	30.91	28.87	**41.93**	39.75	38.40	36.12
	RP4	**32.92**	32.52	30.44	28.47	**40.31**	39.12	37.76	35.52
	RP5	**31.76**	31.48	29.82	28.19	**38.86**	38.19	36.93	35.14
Painter	RP1	38.16	**40.41**	36.78	37.06	45.14	**47.50**	45.34	45.45
	RP2	37.44	**39.44**	36.48	36.69	44.30	**46.37**	44.63	44.65
	RP3	36.13	**37.64**	35.70	35.85	42.82	**44.51**	43.26	43.27
	RP4	35.35	**36.52**	35.08	31.80	41.98	**43.38**	42.36	39.19
	RP5	33.95	**34.74**	33.90	33.90	40.57	**41.62**	40.94	40.84
Frog	RP1	**31.79**	31.61	29.47	30.06	**41.17**	40.84	38.63	39.17
	RP2	**31.43**	31.25	29.27	29.82	**40.79**	40.46	38.37	38.87
	RP3	**30.87**	30.71	28.92	29.44	**40.13**	39.86	37.84	38.35
	RP4	**29.64**	29.49	28.12	28.50	**38.67**	38.39	36.71	37.13
	RP5	**27.88**	27.73	26.79	26.93	**36.43**	36.21	34.98	35.17
Carpark	RP1	**35.13**	35.03	33.55	33.73	43.17	**44.08**	43.19	43.66
	RP2	**34.98**	34.94	33.47	33.66	42.96	**43.88**	43.06	43.46
	RP3	**34.56**	**34.56**	32.99	33.22	42.34	**43.24**	42.32	42.80
	RP4	**34.06**	34.05	25.72	32.49	41.65	**42.54**	34.38	41.74
	RP5	**33.53**	33.37	25.68	31.90	41.07	**41.61**	34.27	40.96
Street	RP1	**37.95**	36.69	35.81	36.00	**46.53**	44.95	46.33	46.13
	RP2	**37.68**	36.43	35.59	35.77	**46.08**	44.45	45.94	45.74
	RP3	**37.46**	36.04	35.34	35.46	**45.67**	43.86	45.46	45.25
	RP4	**36.62**	35.35	34.61	34.62	**44.33**	42.85	44.13	43.88
	RP5	**35.42**	34.33	33.41	33.43	**42.34**	41.21	41.99	41.84
Hall	RP1	40.73	**41.01**	36.91	37.56	48.60	**49.60**	44.26	45.67
	RP2	38.79	**40.87**	36.96	37.54	45.72	**49.15**	44.32	45.59
	RP3	38.51	**40.71**	36.89	37.52	45.43	**48.78**	44.22	45.50
	RP4	38.48	**40.34**	36.85	37.23	45.22	**48.11**	44.15	44.95
	RP5	38.34	**39.52**	36.66	36.91	45.12	**46.73**	43.85	44.30
Mirror	RP1	35.34	**36.51**	34.05	33.26	41.57	**42.91**	40.20	39.45
	RP2	34.05	**35.42**	33.13	32.54	40.21	**41.82**	39.17	38.59
	RP3	32.76	**33.89**	31.96	31.55	38.85	**40.33**	37.79	37.50
	RP4	30.83	**31.48**	30.13	29.84	37.01	**38.02**	35.93	35.84
	RP5	28.49	**28.74**	27.79	27.55	34.36	**34.82**	33.38	33.23
Average	RP1	35.89	**36.17**	33.43	33.30	43.81	**44.06**	41.75	41.65
	RP2	35.20	**35.62**	33.19	33.07	42.93	**43.39**	41.40	41.29
	RP3	34.52	**34.86**	32.72	32.63	42.10	**42.54**	40.72	40.67
	RP4	33.53	**33.78**	31.14	31.39	40.89	**41.30**	38.83	39.20
	RP5	32.20	**32.25**	30.08	30.60	39.24	**39.43**	37.43	37.97

TABLE 5.2: The comparison of average quality of rendered views for different depth estimators used in MIV DSDE, in VMAF and LPIPS metrics.

Sequence	Rate	VMAF (↑)				LPIPS (↓)			
		IVDE	DERS	GA-Net	GwcNet	IVDE	DERS	GA-Net	GwcNet
Fan	RP1	84.71	**85.89**	72.42	74.09	0.085	**0.084**	0.094	0.094
	RP2	84.14	**85.75**	71.98	73.56	**0.089**	0.090	0.099	0.098
	RP3	82.12	**83.10**	70.08	71.46	**0.102**	0.105	0.112	0.111
	RP4	75.61	**76.12**	64.34	65.17	**0.132**	0.138	0.144	0.142
	RP5	**62.15**	61.70	52.92	52.91	**0.166**	0.176	0.179	0.178
Kitchen	RP1	88.81	**89.76**	80.53	76.28	**0.162**	0.163	0.167	0.169
	RP2	87.79	**88.03**	79.50	75.65	**0.190**	0.192	0.193	0.194
	RP3	85.62	**86.11**	77.99	74.71	**0.204**	0.206	0.207	0.208
	RP4	82.23	**82.41**	75.61	72.41	**0.219**	0.221	0.221	0.222
	RP5	**76.70**	76.51	70.44	68.89	**0.229**	**0.229**	0.231	0.232
Painter	RP1	92.74	**94.67**	84.86	85.43	**0.267**	**0.267**	0.275	0.274
	RP2	91.15	**93.08**	84.15	84.94	**0.295**	**0.295**	0.302	0.301
	RP3	87.27	**88.90**	81.80	82.62	0.343	**0.342**	0.348	0.348
	RP4	84.07	**85.12**	79.48	80.15	0.378	**0.377**	0.382	0.382
	RP5	77.63	**78.38**	74.55	74.93	0.405	**0.403**	0.407	0.407
Frog	RP1	**92.58**	91.48	86.45	88.39	**0.067**	0.068	0.070	0.069
	RP2	**91.30**	90.09	85.41	87.28	**0.072**	**0.072**	0.075	0.074
	RP3	**88.94**	87.60	83.54	85.34	**0.079**	0.080	0.083	0.082
	RP4	**83.19**	81.72	78.91	80.29	**0.096**	0.099	0.099	0.099
	RP5	**73.97**	72.45	71.00	71.45	**0.119**	0.120	0.122	0.123
Carpark	RP1	**91.02**	90.77	89.58	89.40	0.173	**0.169**	0.171	0.171
	RP2	**90.74**	90.28	89.29	89.07	0.184	**0.180**	0.181	0.181
	RP3	**89.43**	89.02	87.65	87.54	0.196	**0.193**	0.194	0.194
	RP4	**87.82**	87.25	85.71	85.47	0.202	**0.197**	0.200	0.200
	RP5	**85.97**	85.11	83.50	83.03	0.214	**0.211**	0.213	0.215
Street	RP1	**91.80**	91.13	89.20	88.99	**0.120**	**0.120**	**0.120**	0.121
	RP2	**91.40**	90.57	88.69	88.44	**0.126**	**0.126**	0.127	0.127
	RP3	**90.98**	90.13	87.98	87.65	0.134	**0.133**	0.134	0.135
	RP4	**89.40**	88.42	86.07	85.35	0.150	**0.148**	0.151	0.152
	RP5	**86.28**	85.34	82.57	81.44	0.172	**0.171**	0.173	0.174
Hall	RP1	**92.26**	91.25	90.66	86.66	0.049	**0.048**	**0.048**	**0.048**
	RP2	**91.68**	91.22	90.50	86.55	0.056	**0.053**	0.054	0.055
	RP3	90.92	**91.04**	90.34	86.47	0.065	**0.064**	**0.064**	**0.064**
	RP4	**90.57**	90.22	90.03	86.16	0.077	**0.076**	**0.076**	**0.076**
	RP5	**89.92**	89.78	89.04	85.66	**0.089**	**0.089**	**0.089**	**0.089**
Mirror	RP1	90.24	**91.43**	87.44	85.88	**0.059**	0.060	0.063	0.066
	RP2	87.69	**88.95**	84.79	83.64	**0.074**	0.075	0.077	0.080
	RP3	83.56	**84.26**	80.84	79.55	**0.091**	**0.091**	0.093	0.096
	RP4	74.30	**74.66**	71.69	70.11	**0.116**	0.117	0.118	0.119
	RP5	**59.12**	58.73	56.45	54.63	**0.158**	0.161	0.160	0.162
Average	RP1	90.52	**90.80**	85.14	84.39	0.123	**0.122**	0.126	0.126
	RP2	89.49	**89.75**	84.29	83.64	0.136	**0.135**	0.139	0.139
	RP3	87.35	**87.52**	82.53	81.92	**0.152**	**0.152**	0.154	0.155
	RP4	**83.40**	83.24	78.92	78.14	**0.171**	0.172	0.174	0.174
	RP5	**76.47**	76.00	72.56	71.62	**0.194**	0.195	0.197	0.198

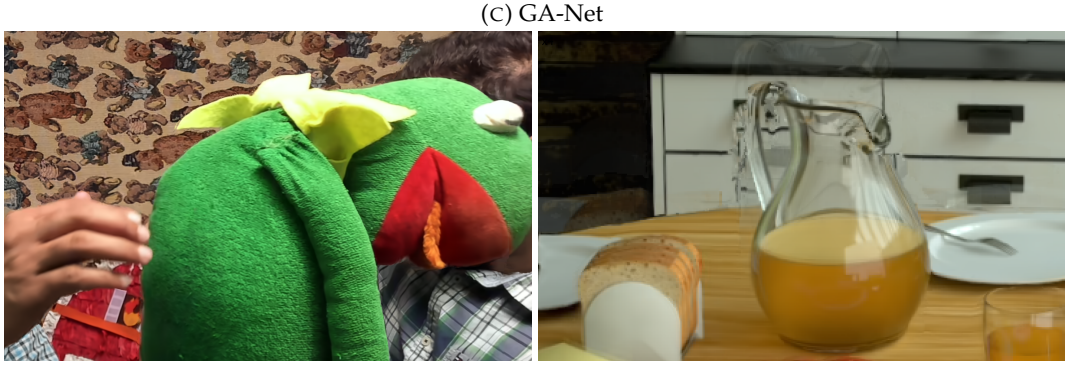
than those for GA-Net and GwcNet, the results for the LPIPS metric indicate that the average quality is very similar for all tested depth estimation methods. What is most important is that, even though these methods were not fine-tuned on the MPEG test set, they present sufficient quality for both indoor and outdoor video sequences and are especially competitive for outdoor sequences (*e.g.* Street and Carpark), as they are more similar to the KITTI driving images.

It is noticeable that the quality of depth maps degrades with the increase in baseline between a pair of views because GA-Net and GwcNet are not optimized for such wide baseline stereo images. Again, this highlights a strong dependency on the sequence properties that are similar to the ones of the training set, causing some lack of robustness for these methods. However, this indicates a very high potential for using deep-learning methods in the DSDE framework, as further improvements are possible by re-training the models on appropriate content, *i.e.* considering multi-view, high-resolution compressed textures as input, or optimization for view synthesis instead of depth fidelity. For most of the sequences, in the case of synthesis using GwcNet depth maps, the compression with increasing QP yields a synthesis quality that is closer to the synthesis quality obtained using IVDE and DERS depth maps. One may conclude that the quality of GwcNet depth maps does not depend as heavily on the quality of transported views as it does in the case of classical methods. However, this behavior is also influenced by the limits of the proposed deep learning approaches for this type of content because the synthesis quality they can achieve for the high-quality transported views is somewhat saturated.

(A) IVDE

(B) DERS

(C) GA-Net



(D) GwcNet

FIGURE 5.1: Subjective comparison for fragments of rendered views.
Left: Frog, right: Kitchen sequence.

When it comes to subjective quality, both types of methods have their advantages and disadvantages. IVDE and DERS depth maps are noisier, but they have sharper object edges, while the two deep-learning-based methods produce depth maps that are somewhat cloudy and have smoother depth discontinuities. As a consequence, the synthesis results based on IVDE and DERS depth maps preserve the object edges

(A) IVDE

(B) DERS

(C) GA-Net

(D) GwcNet

FIGURE 5.2: Subjective comparison for fragments of rendered views for the Carpark sequence. Left: RP1, right: RP5.

better, whereas the results obtained using GA-Net and GwcNet depth maps often have ghosting artifacts around the objects (see a fragment of the Kitchen sequence in Fig. 5.1). On the other hand, in some examples, the deep learning approaches better preserve the consistency of the objects that are uncovered (disoccluded) from one frame to another (see Carpark in Fig. 5.2). The visuals of the Frog sequence in Fig. 5.1

TABLE 5.3: Runtime [s] of decoding and rendering with different depth estimation methods.

Sequence	Method			
	IVDE (CPU)	DERS (CPU)	GA-Net (GPU)	GwcNet (CPU)
Fan	494.27	978.75	78.99	337.58
Kitchen	442.56	992.29	124.00	397.42
Painter	669.04	1186.23	119.67	465.65
Frog	1266.24	1845.84	27.23	358.35
Carpark	385.32	489.35	62.65	264.33
Street	327.05	1033.02	100.98	311.67
Hall	224.21	597.89	29.18	136.99
Mirror	272.33	350.67	74.57	242.10
Average	510.13	934.26	77.16	314.26

show good quality in the case of all depth estimators. Moreover, in Fig. 5.2, we can observe the results for different rate points (RP1 and RP5) and see the degradation in the depth estimation quality caused by severe compression.

The complexity of the MIV DSDE mode when various depth estimate techniques are used is another important factor when conducting experiments. Therefore, to preserve the integrity of the presented results, the runtimes of decoding and rendering are provided in Table 5.3 in order to show the observed range of values when using the presented software. They are not meant to serve as a direct comparison of methods since the estimation for different depth estimators was performed using various computing hardware (of similar but not identical performance). As one can see, GA-Net was the only technique that utilized the GPU, hence it was the fastest. In order to fully assess the quality, it was necessary to display all views (*e.g.*, 25 views for the Kitchen sequence) and estimate all corresponding depth maps, which is what the provided runtimes demonstrate. Since only one desired viewpoint needs to be rendered at a time, the decoding in the MIV DSDE mode is substantially faster in real-world applications. Furthermore, the implementations of the employed depth estimators and MIV decoder were used for academic and standardization purposes rather than being optimized for low computing complexity.

Finally, Table 5.4 shows the comparison of the MIV anchor mode with IVDE-estimated depth maps versus the MIV DSDE mode with GA-Net depth maps. The comparison is done using the BD-rate metric, calculated for the four smallest RPs and the four largest RPs. Here, the sign "$---$" means that there is not enough overlap to compute the BD-rate, and it shows the losses of the MIV DSDE for the Kitchen and Hall sequences. Despite the fact that the objective quality of the rendered views was not the highest for this method, it can be observed that for most of the presented video sequences, on low bitrates, the BD-rate for IV-PSNR indicates a gain in comparison to the MIV anchor mode. This demonstrates once again how effective the MIV DSDE mode is for coding immersive video without heavily relying on a depth estimation technique utilized at the decoder side.

TABLE 5.4: BD-rate [%] of MIV DSDE (GA-Net) vs. MIV anchor (IVDE).

Sequence	Y-PSNR		IV-PSNR		Atlas	Video	Dec
	High-BR	Low-BR	High-BR	Low-BR	Enc	Enc	& Ren
Fan	7.3	-24.0	22.5	-15.0	1.0	80.6	685.7
Kitchen	---	---	---	---	0.6	56.1	837.9
Painter	-62.0	-69.3	-67.3	-70.4	1.2	74.8	1110.8
Frog	-2.2	-28.5	3.3	-25.0	1.2	85.6	553.7
Carpark	102.4	5.7	-7.6	-28.1	1.7	83.2	637.8
Hall	---	---	---	326.5	1.6	204.3	634.0
Street	261.8	64.3	-29.4	-35.7	2.0	85.7	633.1
Mirror	8.0	-32.4	63.9	-18.8	1.0	80.8	900.2

5.3 Conclusion

In this chapter, we presented a study that analyses the performance of different depth estimation tools in the MIV DSDE framework, which is a rare depth estimation comparative study for immersive video use cases and the first to consider the impact of compressed textures on depth estimation and rendering quality. The performance of depth estimators is evaluated under regular MPEG common test conditions on perspective video sequences. We compared two traditional multi-view and two learning-based stereo depth estimation methods, proving the versatility of the MIV DSDE system in regard to the choice of a depth estimator. The objective quality of rendered virtual views is presented on five rate points, computed with various metrics.

Even though learning-based techniques have shown significant progress recently, especially when dealing with complex scenes, some aspects should still be improved in the future. Depth estimation tools for immersive video should be robust to the different nature of the content, compatible with wide-baseline and high-resolution content, arbitrary camera arrangements, and diverse projection formats.

Chapter 6

Patch Decoder-side Depth Estimation in MPEG Immersive Video

In this chapter[1], we investigate the source content pruning as a possible improvement of an immersive video coding system. We explore a concept more challenging than DSDE, which is called patch decoder-side depth estimation. Namely, we propose to go one step further and bypass the transmission of depth patches in the MIV main setting, which considers not only full but also partial views. Although the information transmitted via TMIV is, from a pruning perspective, considered non-redundant, our two proposed schemes show that it is possible to enhance this algorithm. Knowing that depth information is somewhat present in texture videos, we demonstrate that some depths can be omitted and afterward recovered at the decoder side. The first study shows up to 18.4% peak BD-rate reductions on Y-PSNR, decreasing the pixel rate by 8.3% for each test sequence. The second study proposes a reliable recovery of depths at the decoder-side, by introducing patch depth selection at the encoder. This method provides BD-rate improvements on both high and low bitrate ranges, with up to 22.57% Y-PSNR metric gain.

6.1 Introduction

Since bitrate and pixel rate constraints are crucial for an immersive video end-to-end transmission system, there have been a lot of improvements and evolution of TMIV in these aspects. The pruning process performs pixel-matching among all possible pairs of views. It prunes a pixel in one view if it finds a similar pixel in another (which has the same position in 3D space), while it preserves pixels that are not visible in other views [123, 124]. The quality of depth maps has a significant impact on pruning and rendering. Depth maps have a lot of spatial redundancy and it is burdensome to compress them with conventional 2D video codecs. Thus, spatial downsampling of depth maps is enabled in the TMIV processing chain, even though its benefits are counter-intuitive.

Another approach emerged, called decoder-side depth estimation (DSDE) [25, 125], as explained in more detail in Chapter 3. DSDE improved the immersive video system by entirely avoiding the transmission of depth maps and moving the depth estimation process to the decoder-side. The DSDE approach was studied for the case of fully transmitted views (without pruning), where the depth estimation process took advantage of many views that are available at the decoder. However, if

[1]The content of this chapter is based on the work we published in [29] and [30].

one wants to apply the DSDE ideas in the case of the TMIV framework, some difficulties arise, because the number of available views at the decoder side in TMIV is drastically reduced.

The goal of this chapter is to explore the idea of reducing the transmission of depth data in the context of TMIV, and "MIV anchor" mode. More precisely, we investigate the assumption that it is possible to avoid the transmission of some patch depths that originate from the "additional" views while saving the bitrate and pixel rate, and preserving the rendering quality.

The first proposed approach (pDSDE) is "blind" in the sense that it chooses to avoid sending one from a few available patch depth atlases, without any quality-based criteria, thus inspecting the isolated impact of the patch level depth recovery at the decoder side. It yields significant gains over the anchor, especially on low bitrates: we observe an average BD-Rate Y-PSNR gain of 1.8% for medium bitrate (with gains up to 16.0%), and an average of 7.7% for low bitrate (with gains up to 18.4%). Moreover, we achieve 9.3% VMAF [109] BD-Rate reduction for medium bitrate, and 13.4% for low bitrate, while having 6.7% perceptual MS-SSIM [108] gain for medium bitrate, with 11.6% gain for low bitrate. In addition, in all the cases we achieve an 8.3% pixel rate reduction.

The second study proposes an improvement (hpDSDE) compared to the "blind" patch-DSDE approach. We show that it is possible to reliably discriminate between the depth patches that have to be sent and those that can be estimated at the decoder, based on the quality of estimated depth patches at the encoder side. We omit their transmission and subsequently recover them at the decoder side using a state-of-the-art depth estimator. The results show significant gains in comparison to the TMIV-coded anchor: on low bitrate, an average BD-rate gain of 4.63% for Y-PSNR, 6.21% for VMAF, 5.70% for MS-SSIM, and 4.98% for IV-PSNR, and on high bitrate: 2.03% for Y-PSNR, 4.05% for VMAF, 3.15% for MS-SSIM, and 2.28% for IV-PSNR metric.

The chapter is organized as follows. Section 6.2 of this chapter considers the isolated impact of the patch-level depth recovery at the decoder side (patch-DSDE). Section 6.3 of the chapter proposes an improvement compared to the first, "blind" patch-DSDE approach, based on depth patch selection. Section 6.4 draws conclusions on this chapter.

6.2 Omitting the depth patch transmission in MIV

This section gives an overview of our patch-DSDE approach, called pDSDE, where some depth patches are not transmitted, and subsequently, they are estimated at the decoder side. In this section, we will first present the proposed method, and after we will give experimental conditions, results, and their discussion.

6.2.1 Proposed method: pDSDE

In Fig. 6.1, we depict the proposed approach, pDSDE, in comparison to the anchor pipeline: the scheme describes the anchor when the switch is in position 1 and our method when it is in position 0. Note that we consider only the data available at the decoder (receiver), which consists of atlases with "basic views", atlases with patches from "additional views" (here called patch-atlases), and metadata. Consequently, the decoded atlases have compression artifacts. Initially, the process of view *unpacking* is done, as in Fig. 6.2 where each texture patch from chosen patch-atlas is projected

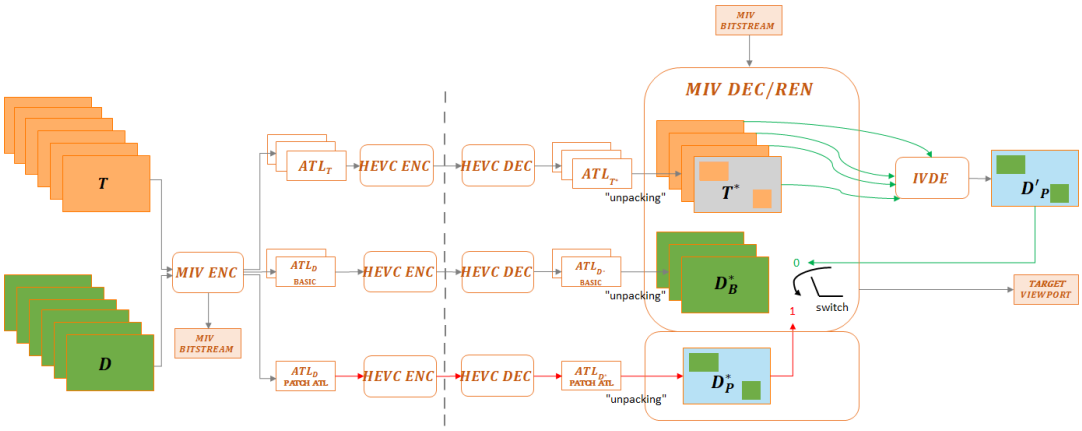


FIGURE 6.1: Process diagram for the anchor ($switch = 1$) and our pDSDE method ($switch = 0$) in the TMIV framework.

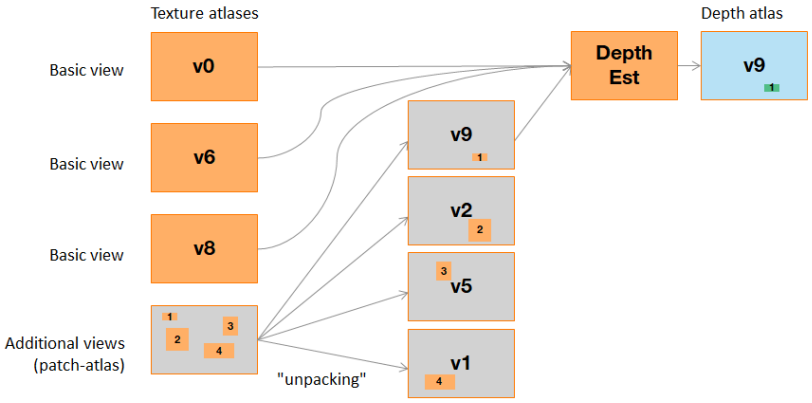


FIGURE 6.2: Unpacking and patch depth estimation.

to the corresponding view. This way, we recover the pruned textures, which we subsequently use in the depth estimation process together with the basic view textures. Following the process of unpacking, the recovered pruned textures and basic view textures are given to the Immersive Video Depth Estimation (IVDE) software [56]. IVDE is a reference software for exploration experiments adopted by MPEG-I, described in Chapter 2. We chose this software because it is agnostic to the number and positioning of the cameras and it ensures high-quality estimated depth maps, with inter-view and temporal consistencies [64]. IVDE performs the depth estimation on segments, which results in correspondence between the object edges in depth maps and the object edges in input textures, consequently enhancing the synthesis quality. No special adaptation was applied to the IVDE software to warn the estimator about the fact that the input textures contain a significant amount of non-valid pixels (those for which no patch has been transmitted).

Let us denote the source textures as $\mathbf{T}$ and source depth maps as $\mathbf{D}$. All texture atlases are transmitted, while for depth atlases we have the following: in the anchor pipeline all six depth atlases are sent, while in the case of our method all three basic depth atlases and two depth patch-atlases are sent, whereas one depth patch-atlas is not sent. Fig. 6.1 is simplified to show only one texture and depth patch-atlas, although there are multiple ones in our setup. At the decoder side, the renderer performs the unpacking: a projection from the atlases to the corresponding positions in the views. Recovered textures are denoted as $\mathbf{T}^*$ and recovered basic view depths

are denoted as $\mathbf{D}_\mathbf{B}^*$. In the anchor case, all depth patch-atlases are sent, and recovered patch depths denoted as $\mathbf{D}_\mathbf{P}^*$, are used in the rendering process. In our case, the decoding process is done in two stages. First, it recovers the textures $\mathbf{T}^*$ and the depth maps $\mathbf{D}_\mathbf{B}^*$. Then, the textures $\mathbf{T}^*$ are given to the IVDE and used to produce patch depths $\mathbf{D}_\mathbf{P}^{'}$. After obtaining the $\mathbf{D}_\mathbf{P}^{'}$, the rendering of the target viewport is continued.

6.2.2 Experimental results and discussion

Test conditions

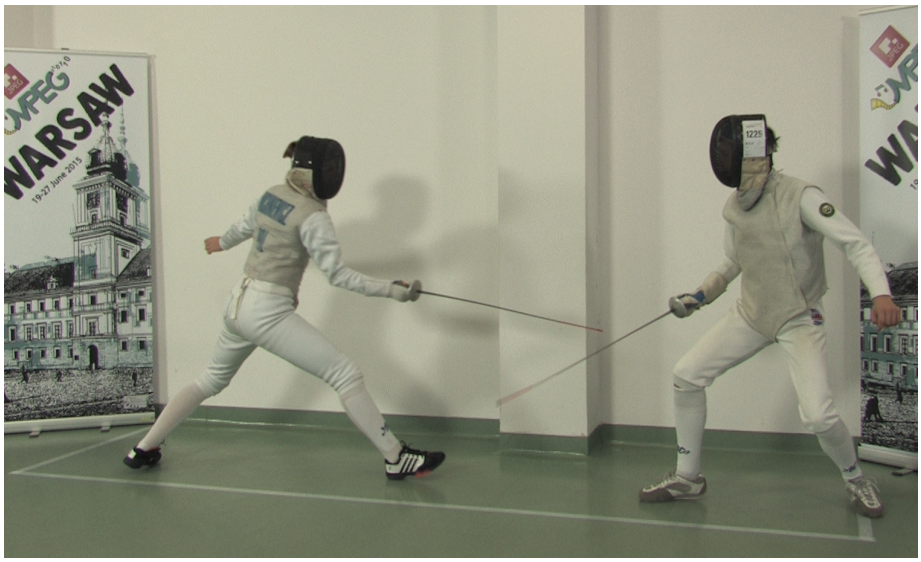
Our experimental setting complies with the MPEG methodology and common test conditions (CTC) defined in January 2020 [101], with TMIV version 4 [92], except for the following: the atlases are constructed with the same size as the source videos, and the depth atlases are not down-sampled. These changes are made to facilitate the experimental process and the comparison with the anchor. Each sequence was encoded in the setup of three groups, where each group has one atlas containing a basic view and one patch atlas. Furthermore, in addition to the five given quantization parameter pairs (QP_T, QP_D) for video compression with HEVC Main 10 profile, another quantization parameter pair is added to show the performance at a low bit rate. The set of (QP_T, QP_D) pairs is: $\{(22,4),(27,7),(32,11),(37,15),(42,20),(47,25)\}$. The first four pairs are used for high bitrate, the second to fifth for medium bitrate, and the last four for low bitrate. Our approach was tested on eight perspective sequences with HD and 2K resolutions, two of which are computer generated (CG), and six of which are natural content (NC). We evaluated the bitrate and synthesized view quality performance provided by the proposed method compared to the anchor, using the Bjøntegaard delta rate metrics, in terms of Y-PSNR, VMAF, and MS-SSIM, as given by the CTC.

Results

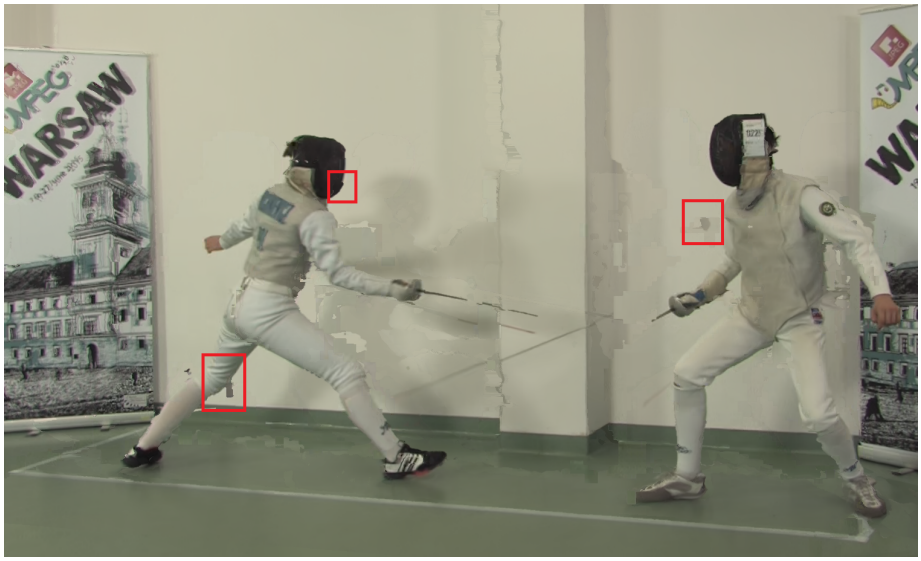
For each sequence and each QP_T value, one patch-atlas with multiple patches was unpacked and saved as a set of the recovered pruned textures. This method was tested for each of the three depth patch-atlases individually, and the one with the best performance was chosen. The corresponding depth atlas was not transmitted and patch depths were replaced with the depths obtained with IVDE from decoded textures per each QP_T, at the decoder side. The obtained results are shown in Table 6.1 and Table 6.2. Negative values indicate BD-rate gains, while positive values indicate losses. The data show BD-rate losses for CG sequences on high and medium bitrate range that diminish with the increase of QP parameters. Moreover, the trend of bigger BD-rate gains as QPs increase is constant for all the sequences. In addition, this method yields an 8.3% pixel rate reduction per sequence.

Fig. 6.3 demonstrates how some artifacts on the Fencing sequence can be avoided with pDSDE by not sending some of the patch depths (marked with red rectangles). Fig. 6.4 shows zoomed details of the Fencing test sequence, encoded at 10 Mbps. In this case, the synthesis result of the pDSDE method subjectively seems better, since it preserves the colors and it has less noticeable artifacts. Fig. 6.5 compares the RD curves of the anchor and proposed method for the Fencing sequence. Our method performs better on the whole bitrate range.

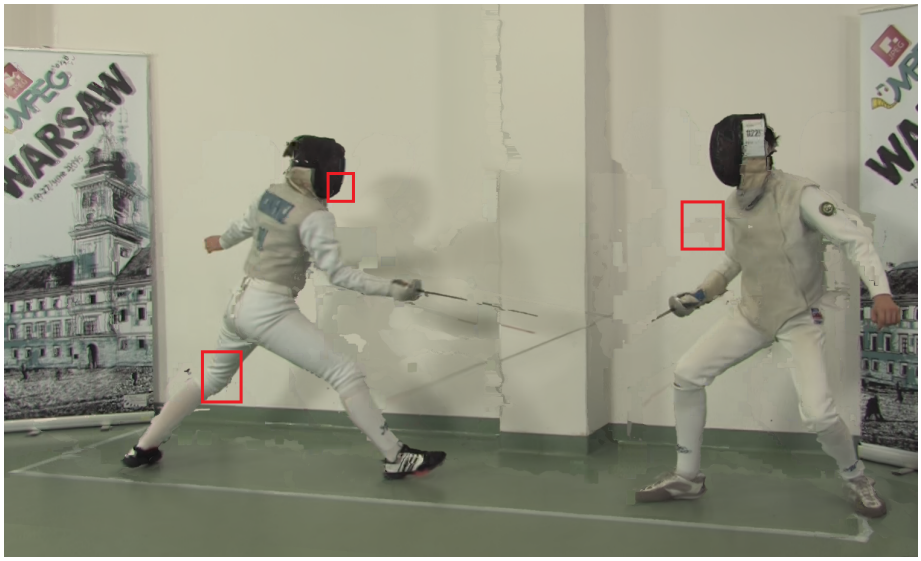
Depending on the type of depth map, a certain percentage of the bitrate, which comprises a texture and depth component, is used. CG sequences, Kitchen and Shaman, as well as NC sequence Frog, have a depth fraction from 30% on high



(A) Source texture.



(B) Synthesized: anchor.



(C) Synthesized: proposed pDSDE.

FIGURE 6.3: One of the views in the Fencing sequence, compared to the anchor synthesis result, and synthesis result of the proposed method.

(A) Source texture.

(B) Synthesized: anchor.

(C) Synthesized: proposed pDSDE.

FIGURE 6.4: Details of the Fencing sequence, compared to the anchor synthesis result, and synthesis result of the proposed method, at 10 Mbps.

TABLE 6.1: BD-rate results per test sequence, in terms of Y-PSNR of synthesized texture [%]. Negative values indicate gains.

Sequence	CTC - High bitrate	CTC - Medium bitrate	Low bitrate
Shaman (CG)	26.34	8.99	0.73
Kitchen (CG)	66.95	30.33	10.72
Painter (NC)	2.75	-7.81	-12.86
Frog (NC)	3.57	-3.49	-8.01
Fencing (NC)	-12.33	-16.02	-18.35
Carpark (NC)	0.26	-8.33	-12.63
Street (NC)	-6.10	-8.67	-10.65
Hall (NC)	-8.53	-9.48	-10.17
Average (all)	**9.11**	**-1.81**	**-7.65**
Average (NC)	**-3.40**	**-8.97**	**-12.11**

TABLE 6.2: BD-rate results per test sequence, in terms of VMAF and MS-SSIM metrics [%]. Negative values indicate gains.

Sequence	VMAF			MS-SSIM		
	High	Med	Low	High	Med	Low
Shaman (CG)	3.44	-6.69	-11.53	20.60	1.54	-6.33
Kitchen (CG)	46.96	12.77	-0.28	20.14	5.08	-3.38
Painter (NC)	-13.04	-18.59	-21.31	-4.21	-13.96	-18.22
Frog (NC)	-3.57	-8.57	-11.13	6.19	-5.49	-10.07
Fencing (NC)	-12.91	-16.87	-19.60	-3.74	-13.29	-17.03
Carpark (NC)	-5.53	-13.14	-16.52	-1.70	-12.27	-16.15
Street (NC)	-7.00	-9.52	-11.32	-6.54	-9.31	-11.29
Hall (NC)	-10.84	-13.35	-15.09	3.93	-5.73	-10.22
Average (all)	**-0.31**	**-9.25**	**-13.35**	**4.33**	**-6.68**	**-11.59**
Average (NC)	**-8.82**	**-13.34**	**-15.83**	**-1.01**	**-10.01**	**-13.83**

bitrates, up to 70% on low bitrates. Other NC sequences take from 50% on high bitrates, up to 85% on low bitrates.

Discussion

In this work, we have shown that it is possible to avoid the transmission of some depth patches, and estimate them at the receiver side, while preserving the quality of synthesized views. The TMIV system is very complex with many interconnected blocks. Despite that, we introduce a new paradigm by avoiding the transmission of a significant amount of depth patches, and we demonstrate BD-rate gains for natural content. We observe that the results are very consistent, for the majority of the natural content. The bitrate reduction is significant, as given in an example in Fig. 6.5. For each sequence, the BD-rate results gradually improve towards low bitrate, which indicates that our method would have a satisfactory streaming performance when network bandwidth is limited. Furthermore, this method provides a pixel rate reduction of 8.3% per sequence, which is very important for some use cases, *e.g.*, streaming on mobile devices.

The losses on the computer-generated content can be explained if we take into account the nature of considered depth maps. Since CG depth maps are ground-truth

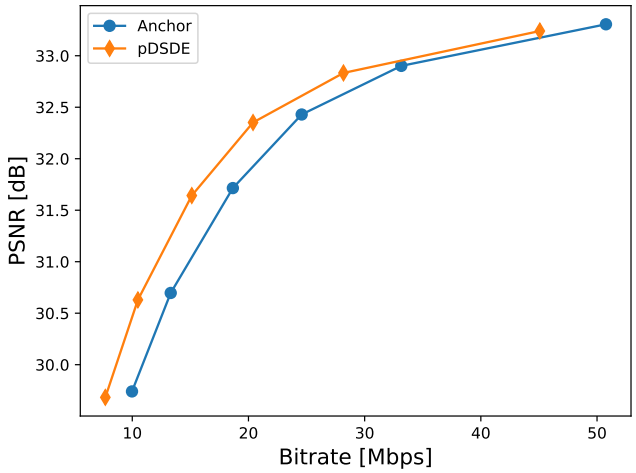


FIGURE 6.5: RD curves for the Fencing sequence.

depth maps, generated with mathematical models of the captured 3D scene, the hypothesis on which pruning is based is perfectly met. Nevertheless, the depth maps for natural content, obtained by a sensor or some depth estimation algorithm, are not perfect, which leaves room for improvement in the pruning method. Some source depth maps in our test set are obtained using IVDE, while others are computed with similar tools, all of which are exposed to additional post-processing methods. In the case of CG content from our test set, the fraction that belongs to depths in the bitstream is significantly lower compared to the depth fraction of the NC sequences. Consequently, the amount of overall saved bitrate for CG content with our method is reduced. Moreover, looking at the patch atlases of the tested CG content, we can notice that the majority of the patches have homogeneous textures, which is very unfavorable for depth estimation algorithms. In addition, our CG content has significantly more source views (25) than the natural content ($9 - 16$). This results in the sparser sampling of texture patches which are preserved in TMIV during pruning. Reduced texture information then leads to deterioration of the quality of estimated depth maps. NC sequences that had BD-rate Y-PSNR losses on high bitrate are Painter, Frog, and Carpark. Despite that, they demonstrated good performance in terms of preserving the perceptual quality, as measured by VMAF in Table 6.2.

Depth estimation is a delicate process that aims to find the correspondence in two or more views, while pruning is built to eliminate the areas which have some correspondence with the other views. Therefore, when we disable the depth patch transmission, we should change the pruning strategy accordingly to ensure a reliable depth estimation at the decoder side. However, in this study, we chose to observe only the isolated impact of the patch depth recovery without modifying the pruning. Aside from the pruning strategy, we are facing more challenges: depth estimation from compressed textures and local depth estimation on very small patch areas. The obtained results are important because they show that the proposed pDSDE method improves the TMIV coding system despite the challenges mentioned above.

6.3 Depth patch selection for DSDE in MIV

This section gives an overview of our hybrid patch-DSDE approach, called hpDSDE, where a reliable, quality-based depth patch selection is done on a per-patch basis at the encoder side. The decision (send / do not send) is transmitted to the decoder

as a flag in the bitstream. In this section, we will first outline the proposed method, and then we will detail experimental conditions and results, and discuss them.

6.3.1 Proposed method: hpDSDE

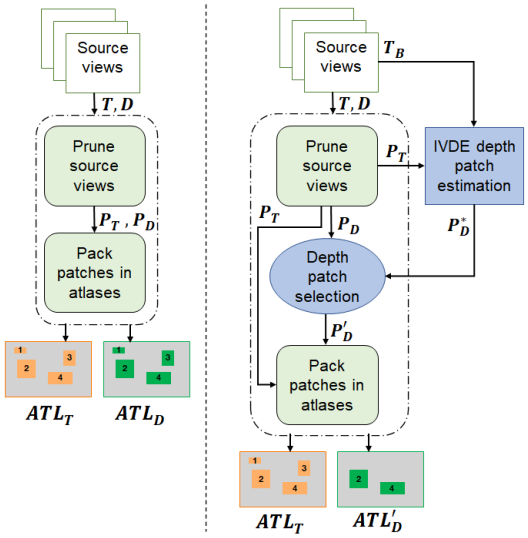


FIGURE 6.6: Left: TMIV anchor encoding of additional views. Right: Proposed method for depth patch selection in additional views.

Depth patch selection

A schematic representation of our hpDSDE method for depth patch selection is shown in Fig. 6.6, as compared to anchor TMIV encoding. This scheme is simplified to illustrate the pruning and packing process in the case of an atlas with patches originating from additional views. First, source views are encoded with the TMIV software. Afterward, the uncompressed basic texture views T_B and pruned "anchor" textures P_T and pruned "anchor" depths P_D views are stored. Then, the depths P_D^* are estimated from the uncompressed textures T_B and P_T. All depth (pruned) views are estimated, except for basic view depths, which are to be transmitted without modifications. The estimated depth patches, obtained from the estimated depths P_D^*, are then compared with the "original" depth patches, obtained from depths P_D, by computing the PSNR: if an estimated depth patch is sufficiently similar to the original one (*i.e.*, the obtained PSNR is larger than a given threshold T_H), we argue that we can avoid sending that patch depth. Only the patch depths that cannot be estimated with sufficient quality need to be sent ($P_D^{'}$ in Fig. 6.6). A flag is stored per patch to convey this information to the modified decoder. The threshold T_H is sequence-dependent since it depends on the quality of the estimated depth patch, which varies across different patches and sequences.

This study is an improvement of the "blind" patch-DSDE (pDSDE) approach explained in Section 6.2: instead of omitting some amount of patch depths without any depth or rendered view guarantee of quality, we ensure that the omitted patch depths are adequate to be estimated at the decoder side. Nevertheless, this approach comes with some inconveniences. First, the transmitted flag is an additional cost to the bitstream, although it is negligible. Besides that, the depth patch selection method is heuristic. Ideally, one should render all possible target viewports using

the estimated depths and compare them with the viewports rendered using the original depths. Since this approach is unfeasible because of its complexity, we propose a suitable proxy (depth quality comparison) as a compromise.

Patch decoder-side depth estimation

The decoding process of the proposed method is similar to the one described in Section 6.2. First, the atlases are decoded and all the transported views are recovered (basic and additional). The TMIV decoder is modified to read the flag which signals if a depth patch is transmitted or not. If a certain depth patch was not transmitted, the process for patch decoder-side depth estimation is invoked at the decoder-side, where the depth patch is estimated with IVDE software from all available decompressed textures: basic views and the corresponding texture patch. The estimated depth patches are written to the corresponding positions of the recovered views. Thus, these estimated depths are used alongside the transported source depths during the rendering process.

6.3.2 Experimental results and discussion

Test conditions

To evaluate our approach and to ensure a fair comparison, we follow the common test conditions (CTC) as defined by MPEG-I [121]. We compare the proposal with the anchor on nine perspective sequences, both natural and computer-generated. In both cases, the depth maps have been generated by IVDE. In our study, we used TMIV7 in the "MIV anchor" mode, which processes the source views (pruning, packing) to generate the atlases. Atlas videos are compressed and decompressed with HEVC (HM16.16), using five different rate points and corresponding quantization parameters *QPs* as defined by the CTC. For decoder-side patch depth estimation in our approach, we used IVDE4 software [56]. To enable evaluation of the results of novel view synthesis using objective metrics, TMIV renders output views in corresponding positions of source views. We evaluated the rendered view quality performance provided by our approach compared to the MIV anchor, with Bjøntegaard delta (BD) rate metrics [103, 104] computed over the four largest *QPs* (low bitrate) and over the four smallest *QPs* (high bitrate). BD-rate is calculated in terms of Y-PSNR, VMAF [109], MS-SSIM [108], and IV-PSNR [126]. The presented synthesis results are averaged over all views, for each *QP*.

Results

The results obtained with hpDSDE as compared to the anchor are shown in Table 6.3a, for the low bitrate range, and in Table 6.3b, for the high bitrate range. Negative values indicate BD-rate gains, whereas positive values indicate losses of the proposed method. The data show average gains on all computed metrics and for both bitrate ranges, with low bitrate peak gains of 22.57% for Y-PSNR, 25.76% for VMAF, 24.07% for MS-SSIM, and 22.94% for IV-PSNR. The peak gains on high bitrate range yield 15.20% for Y-PSNR, 18.70% for VMAF, 13.16% for MS-SSIM, and 12.87% for IV-PSNR. Our method performs particularly well on Frog, Fencing, and Street, while the weakest performance is noticeable on computer-generated content. The fraction of depth maps in the total bitrate is significant, and varies per sequence: 15% to 50% on high and 40% to 80% on low bitrate. On average, our hpDSDE method provides better results for low bitrates, which is coherent with the

TABLE 6.3: BD-rate [%] synthesis results of the hpDSDE proposal vs. anchor, in terms of Y-PSNR, VMAF, MS-SSIM, and IV-PSNR: (a) low bitrate, (b) high bitrate setting. Negative values indicate gains.

(A) Low bitrate synthesis results.

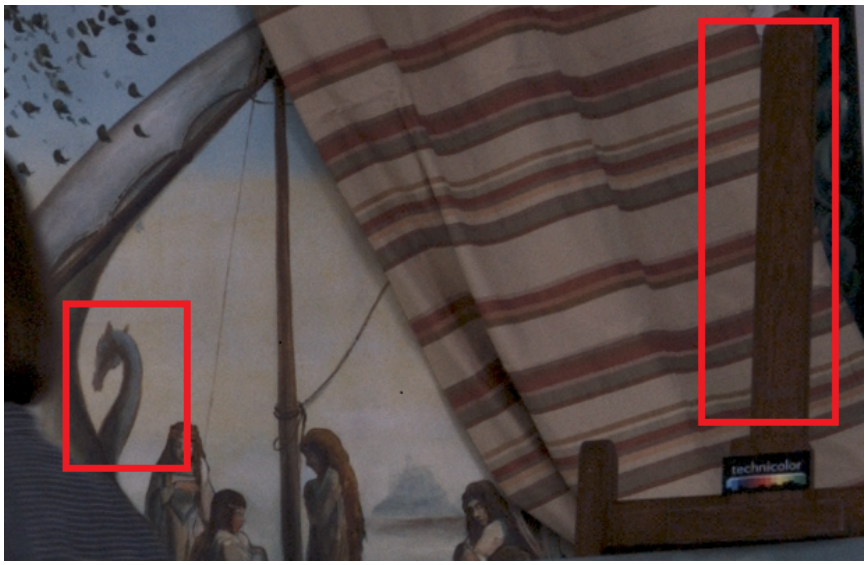
Sequence	BD-rate Y-PSNR	BD-rate VMAF	BD-rate MS-SSIM	BD-rate IV-PSNR
Painter	-0.20	-0.84	-0.66	-0.26
Frog	-9.57	-13.65	-11.60	-9.69
Carpark	-0.18	-0.47	-0.44	-0.16
Fan	-0.14	-0.12	-0.35	-0.02
Kitchen	-0.07	-0.21	-0.08	-0.06
Fencing	-22.57	-25.76	-24.07	-22.94
Hall	0.00	0.00	0.01	-0.04
Street	-8.92	-14.82	-14.09	-11.67
Mirror	0.00	-0.05	0.00	-0.02
Average	**-4.63**	**-6.21**	**-5.70**	**-4.98**

(B) High bitrate synthesis results.

Sequence	BD-rate Y-PSNR	BD-rate VMAF	BD-rate MS-SSIM	BD-rate IV-PSNR
Painter	0.50	-0.73	-0.47	0.51
Frog	-5.02	-11.62	-8.39	-3.58
Carpark	-0.20	-0.42	-0.35	-0.28
Fan	0.36	-0.02	-0.12	0.20
Kitchen	-0.02	0.04	0.00	-0.06
Fencing	-15.20	-18.70	-13.16	-12.87
Hall	-0.12	0.00	0.01	-0.08
Street	1.27	-4.95	-6.06	-4.48
Mirror	0.17	-0.08	0.20	0.12
Average	**-2.03**	**-4.05**	**-3.15**	**-2.28**

results obtained in other MIV DSDE studies [29, 26]. Since legacy 2D video codecs are not convenient for depth map compression, using high compression on them causes harmful artifacts and subsequent deterioration of rendered views. Therefore, hpDSDE approach proves to be beneficial in comparison to the anchor. Furthermore, this method achieves a pixel rate reduction between 0.002% and 5.51% per sequence, which depends on the sequence and the value of its selection threshold T_H (varies from 14 dB to 43 dB). The added flag for depth patch selection gives an average overhead of 0.002% on the high and 0.04% on the low bitrate.

Figures 6.7 and 6.8 depict a visual comparison of the anchor and hpDSDE for the Painter and Street sequences, respectively. As can be seen from these examples, the quality of the rendered views in both cases is very similar. Moreover, selected details of the Painter sequence show that colors and object edges are preserved better in the case of the hpDSDE method. Since the VMAF metric seeks to reflect the viewer's perception of the streaming quality, it is no surprise that our method yields significant gains in terms of this metric.



(A) Source texture.

(B) Synthesized: anchor.

(C) Synthesized: proposed hpDSDE.

FIGURE 6.7: Subjective comparison for fragments of rendered views for the Painting sequence.

(A) Source texture.

(B) Synthesized: anchor.

(C) Synthesized: proposed hpDSDE.

FIGURE 6.8: Subjective comparison for fragments of rendered views for the Street sequence.

The results of our hpDSDE proposal as compared to "pure" patch-DSDE (ppDSDE), instead of the anchor method, are shown in Table 6.4. This comparison aims to show how hpDSDE performs when compared to the method which avoids the transmission of all patch depths. In case of an insufficient overlap of the RD curves, the BD-rate may not be possible to compute, which is indicated with a sign "$---$". In this table, all signs "$---$" indicate gains for hpDSDE proposal. Although PSNR and IV-PSNR show significant gains in favor of the proposal, VMAF and MS-SSIM sometimes show losses, especially on low bitrates. The hpDSDE results for the sequence Street are almost not different from pure patch-DSDE, therefore, we have zeroes in the table.

For a visual comparison, we can observe Fig. 6.9, which shows the synthesis results of the Painter sequence, obtained with ppDSDE and hpDSDE, at 12 Mbps. Moreover, we can see the RD curves for both the Painter and Kitchen sequence in terms of Y-PSNR and VMAF, in Fig. 6.10 and 6.11, respectively. Even though the rate-VMAF curves on low bitrate show better quality for ppDSDE than for hpDSDE in general, after visual inspection of the Painter sequence (Fig. 6.9) it is clear that ppDSDE produces more synthesis artifacts, while hpDSDE produces a frame which is more blurry, since it is compressed with higher QP. The VMAF metric does not account well for the rendering artifacts, since it is not trained for that. At the low bitrate, this behavior is particularly significant. However, on high bitrates, hpDSDE has higher quality than ppDSDE in terms of VMAF.

TABLE 6.4: BD-rate [%] synthesis results of the hpDSDE proposal vs. pure patch-DSDE, in terms of Y-PSNR, VMAF, MS-SSIM, and IV-PSNR: (a) low bitrate, (b) high bitrate setting. Negative values indicate gains.

(A) Low bitrate synthesis results.

Sequence	BD-rate Y-PSNR	BD-rate VMAF	BD-rate MS-SSIM	BD-rate IV-PSNR
Painter	-28.51	57.81	15.92	-13.12
Frog	-2.79	19.59	6.62	-0.99
Carpark	-13.34	13.09	11.23	-2.14
Fan	-51.31	5.75	-3.86	-33.64
Kitchen	---	-75.43	---	---
Fencing	-36.30	-2.85	-42.24	-38.03
Hall	-73.00	-0.60	---	-78.67
Street	0.00	0.00	0.00	0.00
Mirror	-58.06	23.92	-49.46	-55.65

(B) High bitrate synthesis results.

Sequence	BD-rate Y-PSNR	BD-rate VMAF	BD-rate MS-SSIM	BD-rate IV-PSNR
Painter	---	21.21	-45.57	-68.73
Frog	-17.95	10.50	-8.00	-18.75
Carpark	-29.79	-3.75	-35.11	-23.15
Fan	-67.08	-13.91	-28.40	-63.37
Kitchen	---	---	---	---
Fencing	-59.16	-20.55	---	-68.70
Hall	---	-28.02	---	---
Street	0.00	0.00	0.00	0.00
Mirror	-79.33	-23.04	---	---

(A) Synthesized: "pure" patch-DSDE.

(B) Synthesized: proposed hpDSDE.

FIGURE 6.9: Synthesis results of the Painter sequence at 12 Mbps.

Discussion

The main contribution of this chapter is in the reliable discrimination between the depth patches needed for transmission and the ones which are redundant, and therefore, possible to recover at the decoder side. In this study, we have shown that, despite the pruning process, there is some residual redundancy in the set of texture and depth patches. This redundancy can be reduced by omitting the transmission of depth patches that can be accurately estimated at the encoder. More importantly, by imposing an appropriate selection criterion, it is possible to ensure the high quality of estimated depth patches, which consequently results in high-quality rendering.

Threshold: The current version of the method relies on multiple coding passes in order to find the best threshold T_H for patch selection. This results in good rate-distortion performance but also increased complexity. Still, we can reasonably reduce the complexity of these multiple passes. An extensive analysis of the threshold

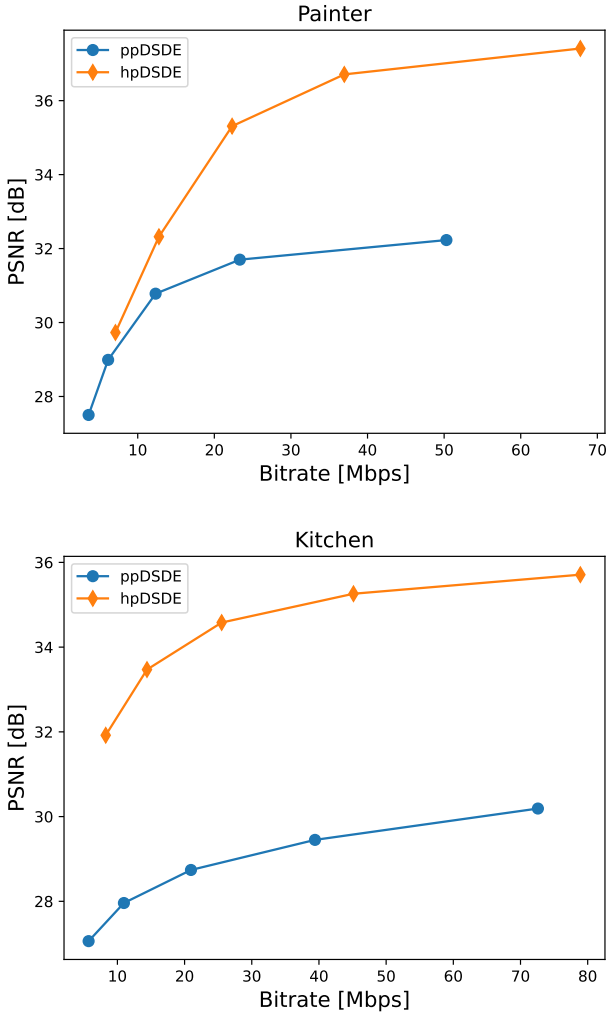


FIGURE 6.10: RD curves for the Painter and Kitchen sequence in terms of Y-PSNR.

per sequence is needed because we do not have an a priori knowledge of estimated depth quality. However, after the first analysis, it is possible to update the threshold solely on an occasional basis, *e.g.,* when characteristics of a sequence significantly change. Thus, this approach is not far from what a practical system could achieve.

Selection criterion: The best criterion for a DIBR method would be to evaluate directly the quality of rendered views obtained using estimated depth maps, as compared to the rendered views obtained using original depth maps. However, since that would be too complex, we propose to use the quality of the estimated depths as a proxy.

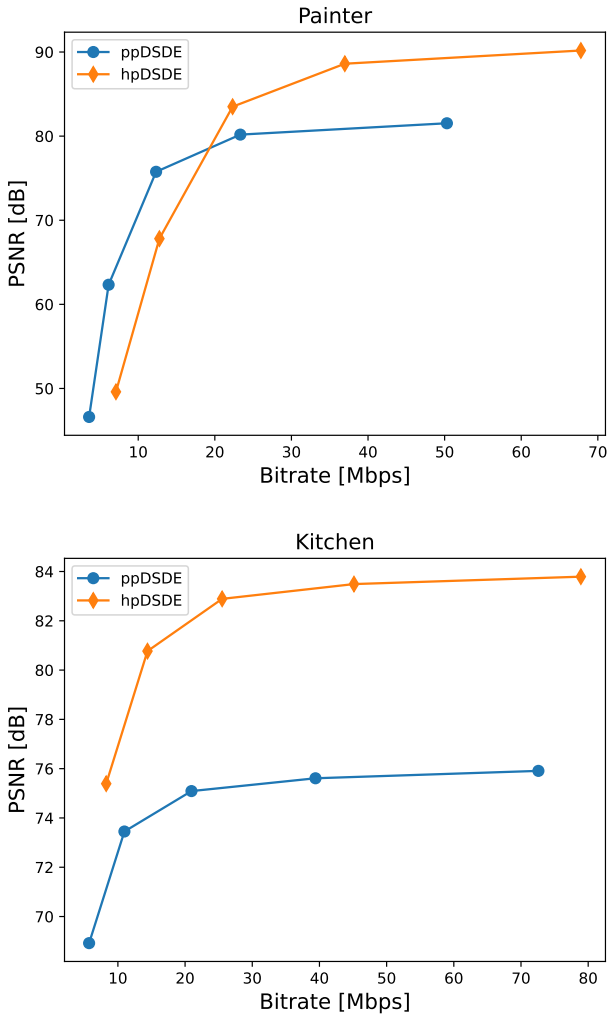


FIGURE 6.11: RD curves for the Painter and Kitchen sequence in terms of VMAF.

6.4 Conclusion

This chapter presented two approaches to tackle the depth patch redundancies and adhere to bitrate and pixel rate constraints in the MPEG immersive video coding setup. More specifically, the first proposal is a proof-of-concept of the idea, which relies on omitting the transmission of some depth patches in TMIV. Consequently, the patch depth estimation is done at the decoder side, using decoded basic views and patches. In the second proposal, a depth patch selection scheme is developed, where the decision for sending a depth patch is made based on the encoder-side depth patch estimation quality, as compared to the source depth patch. Similarly to the first method, the non-transmitted depth patches are recovered using compressed textures at the decoder side. The performance of the proposed methods is evaluated under regular MPEG common test conditions on perspective video sequences. We show BD-rate savings on the whole bitrate range in terms of multiple used metrics, as well as pixel rate savings as compared to the anchor.

Chapter 7

Pixel Pruning in Neural Image-based Rendering

In this chapter[1], we again investigate the source content pruning as a possible improvement of an immersive video system. This time, we focus on novel neural IBR methods, which show impressive performance on complex and non-Lambertian objects and do not require depth maps. Still, the large volume of views necessary to render a virtual view hinders their applications in limited bandwidth environments or prevents their employment in real-time applications. We address this problem with LeHoPP, a method for input pixel pruning, which examines the importance of each input pixel concerning the rendered view, and avoids the use of irrelevant pixels. Even without retraining the image-based rendering network, our approach shows a good trade-off between synthesis quality and pixel rate. When tested in the general neural rendering framework, compared to other pruning baselines, LeHoPP gains between 0.9 dB and 3.6 dB on average.

7.1 Introduction

DIBR methods are not able to handle non-Lambertian content perfectly well because, for such content, the linear hypothesis of the displacement of a pixel in the function of camera displacement is not valid [75]. Therefore, DIBR is rather limited to the view synthesis of diffuse objects. On the other hand, novel neural IBR approaches are able to handle non-Lambertian effects while rendering new views, thanks to their view-dependent nature. Neural Radiance Fields (NeRF) technique [79] has revolutionized the field of neural image-based rendering for complex scenes. However, NeRF-based (implicit) models are impractical due to the lengthy optimization process required to render an unfamiliar scene, since they are encoding a scene into their own weights.

Another method, called IBRNet [73] and detailed in Section 2.5, is a state-of-the-art, robust neural radiance field method. It generalizes well to novel scenes, owing to its end-to-end design that prioritizes the optimization of synthesis quality. Moreover, it outperforms recent one-shot synthesis approaches, like [77], while still having good quality synthesis results as compared to single-scene inference approaches. The current technological progress allows us to have a neural (image-based) renderer on the client side, which produces high-quality synthesis results but requires many available source views of a given scene. This creates a big obstacle to its current deployment using existing standards (such as MIV).

Input pixel pruning has already been pioneered in the context of scalable self-supervised learning in computer vision, where random image patches are masked in

[1]The content of this chapter is based on the work we published in [31].

the input image and the missing pixels are subsequently reconstructed [127]. Moreover, a recent approach proposes an online selection of context points for efficient meta-learning, on 2D videos and other data modalities [128]. Yet, these methods assume that the context (input image) is the same as the target, as opposed to scene rendering, where inputs are different from the target. Regarding multi-view video pixel pruning, TMIV offers a depth-based solution.

In this chapter, we leverage the generalization capabilities of IBRNet for a use-case scenario of immersive video processing and potential data transmission, by taking into consideration the need for a pixel rate reduction. Thus, we propose a method that distinguishes important pixels from the ones that can be pruned. For each input view, our method creates a corresponding pruning mask and removes the non-essential pixels. Pruning mask computation is guided by the loss function computed on target views rendered with IBRNet, with respect to the input views. Then, in the final stage, a target view is rendered using the pruned source views.

Pixel rate is a very important constraint for the feasibility of a multi-view video transmission system, therefore, our work is a significant first stage in the full immersive coding pipeline. We propose a study with random pruning baselines, and we compare them. We observe the better performance of our approach against randomly pruned pixels, and an overall good compromise between rendered view quality and pixel rate, even without retraining and extra fine-tuning per video sequence for IBRNet.[2]

The rest of the chapter is organized as follows. Section 7.2 gives a detailed overview of our proposal, Section 7.3 presents the test conditions and results with their analysis, and Section 7.4 concludes the chapter.

7.2 Proposed method: LeHoPP

This section describes our proposal, called LeHoPP, which is a pruning approach performed at the pixel level of input images, in the image-based rendering setup. Fig. 7.1 provides a scheme of the method. In this section, we will first present how to perform the computation for the pruning mask, and then we will provide an algorithmic overview of LeHoPP. Our approach aims to exploit the existing redundancies among given multi-view images which are used by IBRNet to synthesize a novel view. We hypothesize that it is possible to prune, and therefore, not transmit some input pixels based on their importance factor. During rendering, pruned pixels are recovered in the process of inpainting.

7.2.1 Pruning mask computation

Let us consider the multi-view input images $X_i \in \{0,1,...,255\}^{W_i \times H_i \times 3}$ with $i = \{1,...,N\}$, where N is the number of source views, and W_i and H_i are the width and the height for the i-th view. Before feeding X_i to IBRNet, the images are pre-processed, with standardization and scaling, which transforms X_i to $\hat{X}_i \in \mathbb{R}^{W_i \times H_i \times 3}$, according to the standard implementation of IBRNet [73]. We will associate to every X_i and $\hat{X}_i$ a pruning mask $\mathcal{M}_i \in \{0,1\}^{W_i \times H_i}$. This mask indicates whether some pixels should be either preserved (1) or discarded (0).

To obtain the values of each image mask, we first need to compute an importance score $\mathcal{I}_i$, associated with every pixel of the source view. The importance function tells us how much are rendered images sensitive to changes in a certain pixel of

[2] we used the pre-trained model available at: `https://github.com/googleinterns/IBRNet`

the input image. We know that given the loss function $\mathcal{L}$ we optimize our masks on (mean squared error on the target view, coherently with the loss employed for training IBRNet), we can estimate some variations in the value for the loss function by Taylor series expansion as

$$\Delta\mathcal{L} \approx \frac{\partial\mathcal{L}}{\partial\hat{X}_i(u,v)}\Delta\hat{X}_i(u,v). \tag{7.1}$$

From this, we can define our importance score as

$$\mathcal{I}_i(u,v) = \left|\frac{\partial\mathcal{L}}{\partial\hat{X}_i(u,v)}\right| \cdot \left|\hat{X}_i(u,v) - \hat{X}_i^{\text{inp}}(u,v)\right|, \tag{7.2}$$

where u and v denote pixel coordinates in an image, and $\hat{X}_i^{\text{inp}}$ denotes an inpainted approximation of $\hat{X}_i$. To maximize the performance of the pruning mask $\mathcal{M}_i$, the source importance score $\mathcal{I}_i$ is cumulated over image (R, G, B) channels and averaged over all the target views. Since different subsets of nearby source views are used to render the target views, we average the gradient values that we get for a particular source view but from different target loss computations.

Unlike similar approaches in the literature [129, 130], given that masked values are discarded pixels, we can run an inpainting algorithm to partially recover lost information. Hence, our $\Delta\hat{X}_i(u,v)$ is not simply the value of the pixel (as traditionally done in any pruning algorithm), but it is the distance between the real value and the estimation of its inpainted value. For computational efficiency, we estimate it by averaging the value of neighboring pixels:

$$\hat{X}_i^{\text{inp}}(u,v) = \frac{1}{8}\left[-\hat{X}_i(u,v) + \sum_{i=u-1}^{u+1}\sum_{j=v-1}^{v+1}\hat{X}_i(i,j)\right]. \tag{7.3}$$

Here, inpainting is not used to fill out the occluded regions in the images after the novel view rendering, rather it is used after the pixel pruning because IBRNet is trained on full images. Instead of giving the black/gray pruned pixels to the IBRNet renderer, we are inpainting the pruned regions and incorporating their impact in our importance score.

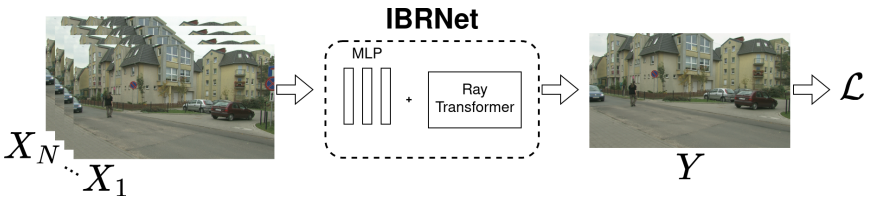
We use the same mask for all the frames of the intra-period, to reduce the frequency of mask updating and overhead for potential transmission. Therefore, the importance score is cumulated over all the frames of one intra-period. Finally, given that we target the remotion of the $\gamma \in [0;1]$ fraction of pixels from the input view, we can use the quantile function $\mathcal{Q}_{\mathcal{I}}(\gamma)$ to determine the threshold to apply to importance values, to finally compute the pruning mask:

$$\mathcal{M}_i(u,v) = \begin{cases} 1 & \text{if } \mathcal{I}_i(u,v,c) \geq \mathcal{Q}_{\mathcal{I}}(\gamma), \\ 0 & \text{otherwise.} \end{cases} \tag{7.4}$$

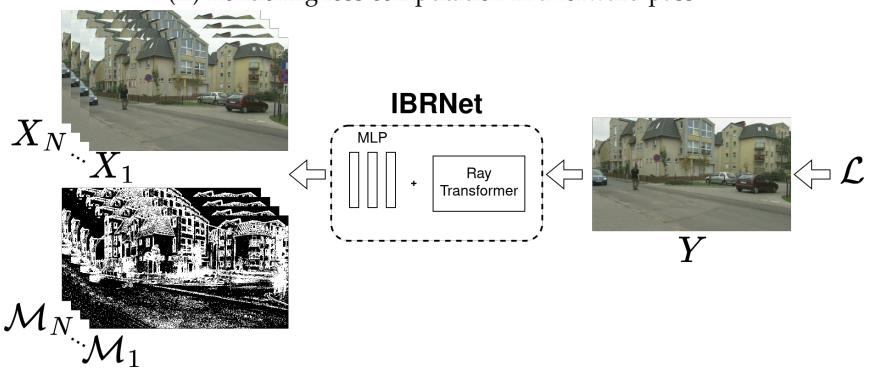
In the next subsection, an overview of LeHoPP will be provided.

7.2.2 Overview of LeHoPP pipeline

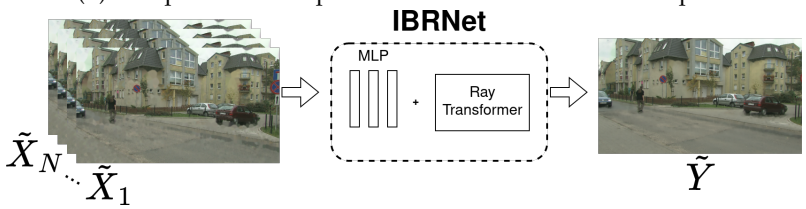
The scheme of LeHoPP is depicted in Fig. 7.1, and it consists of three steps.



(A) Rendering loss computation in a forward pass.



(B) Computation of importance and mask in a backward pass.



(C) Rendering result, from pruned and inpainted images.

FIGURE 7.1: Overview on the LeHoPP method application.

First, the multi-view images X_i are given to IBRNet and a forward propagation step is performed, and the loss $\mathcal{L}$ is computed from the synthesized view Y (Fig. 7.1a).

Then, through back-propagation, we can calculate $\frac{\partial \mathcal{L}}{\partial \tilde{X}_i}$, from these the importance scores $\mathcal{I}_i$ according to (7.2) and the masks are computed according to (7.4) (Fig. 7.1b).

Finally, the masks are applied on the input images to prune the pixels, which are then inpainted, and the resulting images $\tilde{X}_i$ are given to the IBRNet renderer, to synthesize the target view $\tilde{Y}$ (Fig. 7.1c).

In the next section, we will present the empirical validation for LeHoPP.

7.3 Results and discussion

7.3.1 Test conditions

Our dataset consists of eight perspective sequences from MIV CTC [121], with natural and computer-generated multi-view content of high resolution, captured by a sparse camera setup.

We compare LeHoPP with two baselines: *base1* (32×32 block-based random pixel pruning) and *base2* (4×4 block-based random pixel pruning).

Since we did not find any existing method for addressing the problem of pixel pruning as in our context, we decided to compare the proposed method with a simple, "blind" approach (random pruning). We remark that the TMIV pruner is not a relevant comparison in our setup, since it is designed for a different goal. It is

based on depth image-based rendering - while IBRNet deals with texture-only data - and yields a small set of patches, which cannot be used in the IBRNet rendering context. Moreover, the texture videos pruned with our method cannot be utilized in the TMIV rendering, because we do not rely on depth maps.

In our experiments, the pruning percentage γ is the only varying parameter, demonstrating the trade-off between quality and pixel rate. Furthermore, pruned pixels are inpainted prior to being input to the rendering, using Telea [131] method from the OpenCV library. The number of input source views for rendering is set to 9, for all the sequences. We evaluate the quality of synthesized views using PSNR, SSIM [107], and LPIPS [110] metrics, as proposed for IBRNet.

7.3.2 Experimental results

Results with different pruning percentages γ of 5%, 10%, and 20% are shown in Tab. 7.1. They represent the rendered view quality for the target video frames, averaged among all views of a sequence. We can observe that, on average, LeHoPP outperforms both *base1* and *base2* for all the computed metrics, for all three given values of γ. We further notice that, even though γ's value of 20% significantly reduces the number of source pixels in the rendering process, the performance is not very far from the anchor, where rendering is done on the source input without any pruning. Our method performs particularly well on sequences Carpark, Fan, Hall, Street, and Mirror. However, we observe slightly lower effectiveness in the case of sequences Painter, Frog, and Shaman. Some elements of suboptimality that justify this behavior (for the sake of computational efficiency) can be found, for example, in the inpainting approximation used for the importance computation, introduced in (7.3), or in the one-shot nature of the pruning process itself (making multiple smaller steps would make the approximation in (7.1) more precise) and, most importantly, not fine-tuning IBRNet to be more robust to the pruned pixels.

TABLE 7.1: Results for different pixel pruning percentages: 5%, 10%, and 20%.

Metric	γ	Method	Sequences								
			Painter	Frog	Carpark	Fan	Shaman	Hall	Street	Mirror	**Average**
PSNR (↑)	0%	Anchor	31.14	28.18	38.96	37.67	47.06	39.85	43.36	33.38	37.45
	5%	*base1*	29.89	27.25	35.79	33.62	41.41	38.11	37.78	30.43	34.29
		base2	30.72	27.75	37.40	35.17	43.42	39.23	40.49	31.64	35.73
		LeHoPP	**31.04**	**27.98**	**38.76**	**37.40**	**44.76**	**39.78**	**42.73**	**33.13**	**36.95**
	10%	*base1*	29.03	26.43	33.62	31.38	38.58	36.78	35.01	29.08	32.49
		base2	30.31	**27.30**	36.05	33.57	41.29	38.65	38.58	30.57	34.54
		LeHoPP	**30.44**	27.27	**38.45**	**36.76**	**41.74**	**39.69**	**42.10**	**32.37**	**36.10**
	20%	*base1*	27.74	24.96	30.31	28.57	35.36	34.80	31.57	27.22	30.07
		base2	**29.48**	**26.40**	33.78	31.21	**38.46**	37.49	35.78	29.05	32.71
		LeHoPP	28.04	24.90	**37.39**	**34.26**	34.55	**39.42**	**40.37**	**29.75**	**33.58**
SSIM (↑)	0%	Anchor	0.9561	0.9092	0.9901	0.9926	0.9982	0.9965	0.9953	0.9684	0.9758
	5%	*base1*	0.9450	0.8941	0.9788	0.9784	0.9911	0.9934	0.9853	0.9538	0.9651
		base2	0.9484	0.8962	0.9819	0.9819	0.9930	0.9948	0.9881	0.9575	0.9677
		LeHoPP	**0.9511**	**0.9038**	**0.9878**	**0.9913**	**0.9952**	**0.9963**	**0.9930**	**0.9675**	**0.9733**
	10%	*base1*	0.9337	0.8771	0.9652	0.9626	0.9827	0.9914	0.9739	0.9388	0.9532
		base2	0.9397	0.8820	0.9718	0.9695	0.9869	0.9929	0.9797	0.9455	0.9585
		LeHoPP	**0.9412**	**0.8923**	**0.9844**	**0.9872**	**0.9871**	**0.9961**	**0.9894**	**0.9640**	**0.9677**
	20%	*base1*	0.9103	0.8390	0.9333	0.9272	0.9641	0.9855	0.9473	0.9069	0.9267
		base2	**0.9197**	0.8505	0.9468	0.9403	**0.9722**	0.9883	0.9594	0.9180	0.9369
		LeHoPP	0.9061	**0.8509**	**0.9745**	**0.9657**	0.9473	**0.9956**	**0.9791**	**0.9436**	**0.9453**
LPIPS (↓)	0%	Anchor	0.1260	0.1637	0.0694	0.0313	0.0162	0.1167	0.0559	0.0796	0.0823
	5%	*base1*	0.1548	0.1816	0.0883	0.0591	0.0458	0.1335	0.0805	0.1102	0.1067
		base2	**0.1487**	0.1756	0.0831	0.0561	**0.0364**	0.1352	0.0719	0.1046	0.1015
		LeHoPP	0.1496	**0.1743**	**0.0786**	**0.0423**	0.0536	**0.1221**	**0.0667**	**0.0927**	**0.0975**
	10%	*base1*	0.1834	0.2018	0.1108	0.0897	0.0779	0.1519	0.1073	0.1404	0.1329
		base2	**0.1749**	**0.1909**	0.1008	0.0845	**0.0606**	0.1534	0.0916	0.1323	0.1236
		LeHoPP	0.1853	0.1946	**0.0909**	**0.0628**	0.1077	**0.1296**	**0.0813**	**0.1136**	**0.1207**
	20%	*base1*	0.2397	0.2471	0.1627	0.1547	0.1434	0.1908	0.1648	0.2007	0.1878
		base2	**0.2319**	**0.2286**	0.1446	0.1448	**0.1167**	0.1910	0.1388	0.1920	**0.1736**
		LeHoPP	0.2781	0.2544	**0.1242**	**0.1329**	0.2517	**0.1483**	**0.1189**	**0.1756**	0.1855

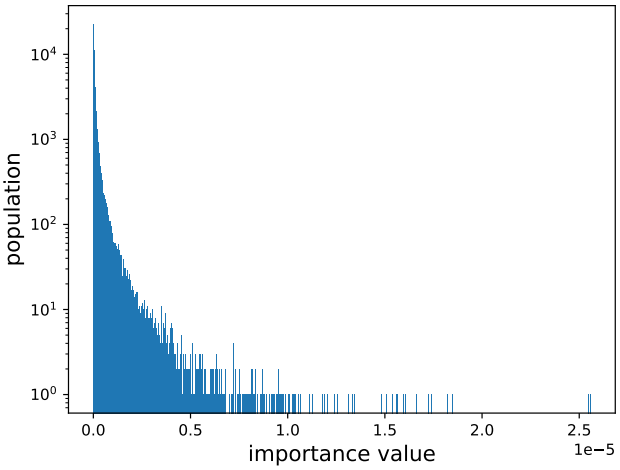


FIGURE 7.2: Histogram of the importance values, Street video sequence, view 1 at frame 0.

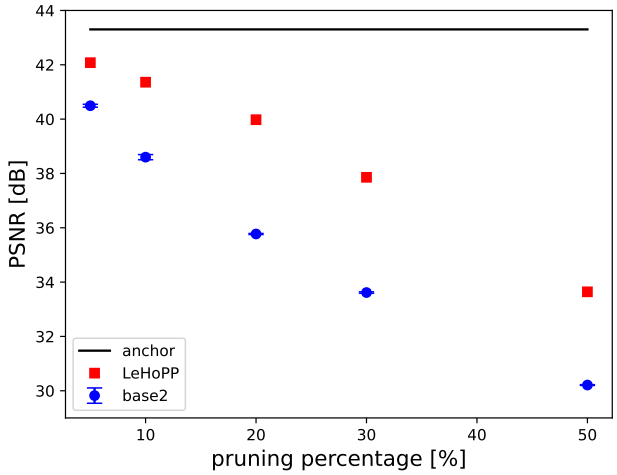


FIGURE 7.3: Rendering results for different random seeds of the method *base2* for Street video sequence.

7.3.3 Analysis and discussion

The main contribution of this work, pixel importance computation, has proven to be a reliable metric for pixel pruning. Let us observe closely the contribution of the key elements of LeHoPP and how consistently it is better than the considered baselines, on a sample scene: we provide here an in-depth analysis using the Street sequence.

Fig. 7.2 shows the distribution of the importance values $\mathcal{I}_i$: the majority of pixels (population) have a very small importance value, showing that it is, in principle, possible to prune a large number of pixels.

Furthermore, let us compare LeHoPP with *base2* for different γ values, averaged and examined on five different seeds and initial random states. Fig. 7.3 shows that the results obtained with the random method are consistently performing worse than results obtained with LeHoPP, independently from the seed given to the random pruning method. We show this analysis on *base2* because it is performing significantly better than *base1*, since its configuration makes it easier to inpaint pruned blocks with higher fidelity.

Additionally, to measure the performance of LeHoPP compared to *base2* in terms

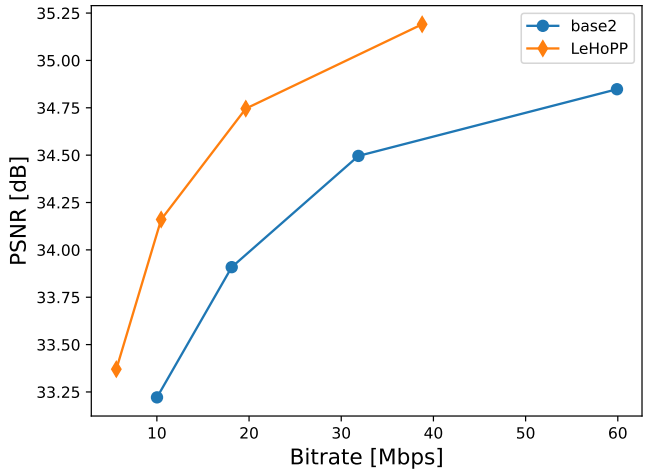


FIGURE 7.4: RD performance of the LeHoPP method as compared to *base2* on Street video sequence.

of data rate, we conduct a compression study. The setup of this study is HEVC simulcast, with QP $\in \{22, 27, 32, 37\}$. First, all views of the Street sequence are pruned with $\gamma = 10\%$; then they are compressed, and finally the decoded pruned views are given to the IBRNet renderer, without inpainting. LeHoPP yields 54.24% BD-rate [103] against *base2*, and performs consistently better in both quality and bitrate, for a given QP (Fig. 7.4).

Replacing the pruned pixels with their inpainted counterparts before rendering helps the network to renderer the views with higher quality. Fig. 7.5 and 7.6 compare the quality of synthesized views of the Street sequence, one without (Fig. 7.5b) and another with inpainting prior to rendering (Fig. 7.6b).

Synthesis results of the Street sequence (without inpainting), after pruning with *base2* and LeHoPP, and compression at 10 Mbps, are shown in Fig. 7.7. We can see that even with compressed content, IBRNet achieves great rendering performance, which demonstrates its robustness. On the rendered view, it is noticeable that LeHoPP pruning creates less annoying artifacts than *base2*, and performs more "structured" pruning.

Pixel rate in the MIV standard [20] is limited to 32 Megapixels at 30 fps. Sequences with high resolution and a large number of source views are barely fulfilling this requirement, if we consider solely the transmission of the source texture views without their corresponding depth maps. Therefore, pruning percentages between 5% and 20%, tested in the scope of this work, are well-fitting to show the real possible use case scenario, where the amount of preserved pixels stays in the defined pixel rate boundaries.

The inpainting approximation that we use for the computation of importance score is a proxy for inpainting, and a good simplification in the case when pruned areas are smaller. Therefore, results in Tab. 7.1 show higher gains of LeHoPP compared to *base1* and *base2* on smaller γ values. Furthermore, during importance computation, fewer views were used as a source input for rendering of Shaman, Painter, Frog, Mirror, and Fan, due to memory limitations and big backpropagation computational graph. This might also result in less precise importance values for these sequences. Moreover, for importance values, the method requires a few minutes of computation per frame, for a specific target view.

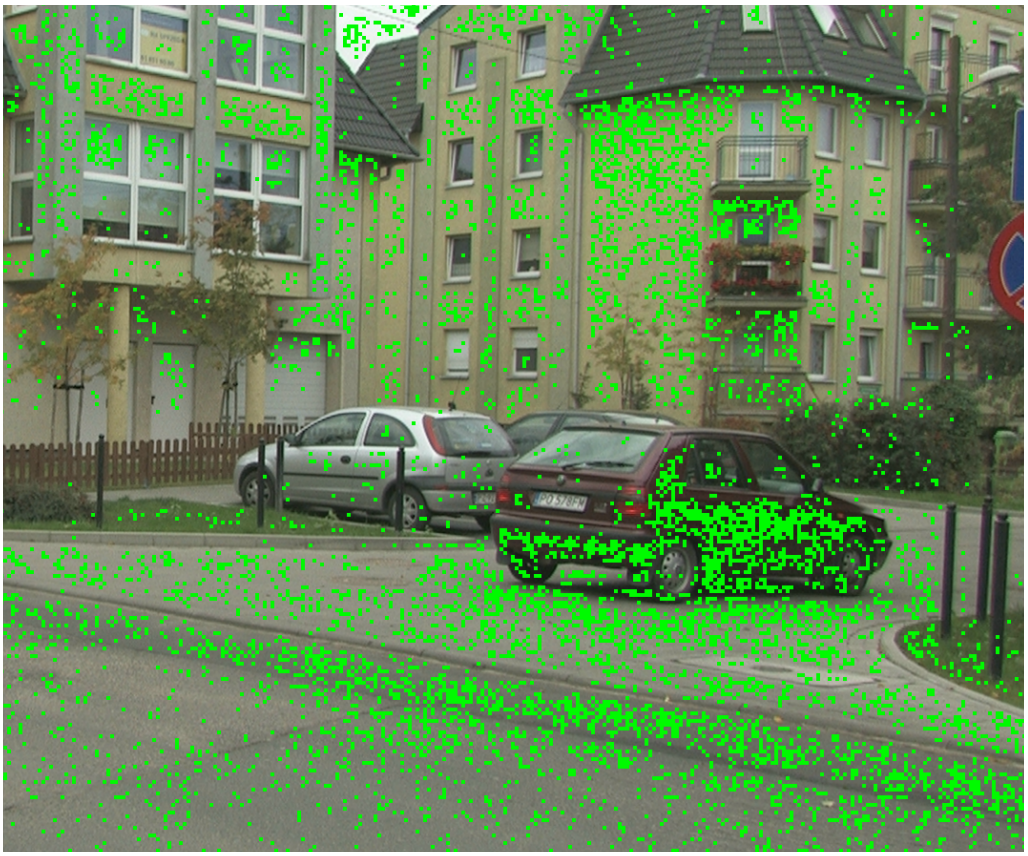
The TMIV pruning technique depends on both the depth maps and textures, and it is created to optimize for DIBR. Moreover, the MIV standard cannot transmit all

the source views of a captured scene. Thus, it is incompatible with a neural renderer such as IBRNet, even though they are compatible in terms of data: the GA profile of MIV can transmit multi-view texture-only videos. Hence, there is a need to prune the textures for the IBRNet renderer for a potential transmission use case. The proposed system is not a standalone immersive video coding setup, but a first, necessary pre-processing stage of such a system. Our study demonstrates the method that learns which pixels are not essential and reduces the pixel rate while maintaining a high synthesis quality.

To summarize, IBRNet showed extraordinary robustness when dealing with non-ground-truth pixels chosen with LeHoPP. The results obtained on sparse camera setup, without retraining or fine-tuning, given some amount of inpainted pixels together with source pixels, are of very good quality, as shown with PSNR, SSIM, and LPIPS objective metrics.

7.4 Conclusion

This chapter presented an approach that learns how to prune the pixels and can be utilized with neural image-based synthesizers in a multi-view setup without retraining the model. LeHoPP leverages the existing redundancy among different views of a scene and uses IBRNet to compute rendered view loss and its impact on each input pixel. Thereby, it is computing the pruning mask for each view, given the desired pruning percentage. We show that neural synthesis itself provides the information needed to prune the content. Our method outperforms random block-based pruning baselines and demonstrates good efficiency on the MPEG immersive video dataset without scene fine-tuning. We believe that our work encourages further development of neural renderers and broader adoption of immersive video standards, such as MIV.



(A) Pruned input without inpainting ($\gamma = 20\%$)

(B) Output without input inpainting (21.35 dB)

FIGURE 7.5: Qualitative comparison of a target image for a given input view: pruned pixels in (a) are in green for visualization.

(A) Pruned input with inpainting ($\gamma = 20\%$)

(B) Output with input inpainting (40.73 dB)

FIGURE 7.6: Qualitative comparison of a target image for a given input view: pruned pixels in (a) are in green for visualization.

(A) Pruned with *base2*.

(B) Pruned with LeHoPP.

FIGURE 7.7: Synthesis results of the Street sequence, after pruning and compression at 10 Mbps, without inpainting.

Chapter 8

Conclusion

Throughout this thesis, we have been working on improvements for a video-based immersive coding system. In this chapter, we give a summary of the research carried out in the scope of this thesis and describe the future perspectives for immersive video coding research.

8.1 Summary and discussion

In Chapter 2 of this thesis, we introduce the reader to the historical background of video-based coding solutions. We detail their benefits in terms of compression efficiency and draw attention to their drawbacks in terms of complexity. In addition, we continue by introducing a new immersive video coding paradigm, which operates on the source material and pre-processes it, whilst it relies on the legacy 2D video codecs to compress the resulting data. We finish by describing the two important (non-normative) stages this system entails: depth estimation and view synthesis. Furthermore, in Chapter 3 we describe the MIV standard, which is the main subject of this research, the algorithms employed in its reference software, and the test conditions followed during our experiments.

Since the previous immersive video coding standards (such as MV-HEVC and 3D-HEVC) were complex and not widely deployed, the MIV standard turned towards the aforementioned novel paradigm for immersive video coding, which is simpler and easier for wide deployment. The objectives of this thesis have been aligned with the standardization of MIV, and aimed at increasing the view synthesis quality and compression efficiency.

In the following chapters, we gave a detailed overview of our studies and the benefits they brought to the immersive video coding system. We studied the areas that may provide improvement to the overall design of immersive video coding: atlas compression, depth estimation techniques, and pruning of the source content. The first potential advancements we considered are related to the codec-agnosticism characteristic of the MIV standard and to the compression of the atlases. We give the related analysis and the results in Chapter 4. The second area of our research for possible advancements is related to the depth estimation techniques used on the multi-view content, and their influence on the compression performance, as well as the synthesized view quality. We give the experimental results and analysis in Chapter 5, and the related depth estimation study is also provided in brief in Appendix A.

The two mentioned areas of research gave us the following insights. In Chapter 4, we confirm that MIV is versatile in terms of using 2D video codecs for atlas compression. We observe that, when depth maps are compressed with legacy codecs that do not implement specialized coding tools such as inter-component techniques, the rate-distortion performance suffers a non-negligible degradation. Our tests show the benefits which are brought by employing the VVC screen content coding tools, even

without any modifications to them. In addition, in Chapter 5, we observe that the conventional depth estimation tools used by MPEG are competitive to neural-based approaches. Both mentioned families of depth estimators have their advantages and disadvantages, and neither brings significantly greater improvements in visual quality. Thus, these two studies have shown a limited impact on the efficiency of MIV, so we continued our research toward the third direction, which is input pruning.

Pruning in MIV is a process responsible for removing redundancies among the source views. That is a new approach that exploits the inter-view redundancies by "compressing" the input data with pruning, and decides on pruning based on view re-projection. However, pruning depends on the chosen depth-image-based rendering (DIBR) methods, and also on depth maps. Moreover, DIBR methods also depend on depth maps, which can be ground-truth or estimated. In the former case, their fidelity is bounded by their resolution or mathematical model, whilst the latter exhibit estimation errors, especially on homogeneous image areas and on non-Lambertian surfaces. Hence, pruning is not perfectly eliminating all the redundancies.

Therefore, in Chapter 6, we examine the possibility of improving the MIV system, by leveraging the existing redundancies after pruning and the decoder-side depth estimation (DSDE) paradigm. DSDE was previously introduced and tested solely on full views and in the presence of a big number of available views for depth estimation. We demonstrate that the notion of DSDE can be applied for a more challenging scenario - in the case of small image areas called patches, and the low number of available full views. We first introduce the pDSDE approach, which consists in bypassing the transmission of a certain amount of patch depths and retrieving them at the decoder side, prior to rendering. The pDSDE method has shown good results on natural content and especially good performance on low bitrates. These results are consistent with the results obtained with basic, full-view MIV DSDE configuration. However, during MIV standardization, the pruning techniques implemented in TMIV were further improved. This has driven our research towards the more reliable, hybrid selection for patch-DSDE, called hpDSDE. This approach examines the quality of the estimated patch depth at the encoder side and avoids its transmission only if its quality is sufficiently high, as compared to the source patch depth. The hpDSDE method has shown especially good results on natural content, and some improvements on computer-generated content. Both pDSDE and hpDSDE ensure fewer visual artifacts induced by harmful compression of depth maps, and higher rendering quality for the same bitrate, for the majority of test sequences.

Finally, in Chapter 7 we go a step further and examine the impact of input (texture) pruning on one of the novel image-based rendering methods based on NeRF, called IBRNet. We have chosen IBRNet since it removes the burden of depth capture and estimation, and it provides an impressive visual quality when rendering complex, non-Lambertian scenes. Moreover, it is a general rendering method that performs well on unseen scenes, which alleviates the need for per-sequence retraining or fine-tuning. Our results show that it is possible to learn how to prune the input images while ensuring good performance in terms of rendering quality. In fact, we show that the neural synthesis itself provides the information needed to prune the content, which is a truly novel approach for identifying redundant pixels.

8.2 Future perspectives

Here we present the perspectives for different areas of future research that have been opened by work conducted in this thesis. Although our main contributions

demonstrated notable advancements in their respective fields, we have identified some limitations and potential opportunities for further exploration.

8.2.1 Patch decoder-side depth estimation

The work done in the scope of this thesis (Chapter 6) has shown some possible future applications for the GA profile. Even after the first edition of the MIV standard and its GA profile, the work on improvements for a DSDE system has continued [132]. At the time of writing of this manuscript, a novel version of the GA profile (DSDE) for MIV Ed.2 is proposed [133], which aims to allow including partial or complete geometry and occupancy information in the bitstream. This proposal for the DSDE profile is inspired by many potential use cases, including our pDSDE work [29]. The authors [132] claim that our pDSDE scheme could be enhanced by the use of external occupancy video. This would signal which pixels are pruned and which are preserved, and ensure the spatio-temporal redundancy removal by masking the pruned pixels, thus saving the bitrate [134]. Another potential use case of the improved DSDE profile is to signal depth refinement (post-processing) [135] as a form of decoder-side depth estimation. This could be especially beneficial for the synthesis quality on low bitrates.

Not transmitting some patch depths brings new possibilities, since it saves the pixel rate previously reserved for these patch depths. For example, the rendering could benefit from the transmission of some additional information in the atlases instead of patch depths. The additional information can be given in the form of added texture which was initially pruned, or another kind of attribute, like transparency.

Indeed, moving towards decoder-side depth estimation brings additional complexity to the decoder. However, the MIV standards support Geometry Assistance SEI, which enables using a set of encoder-derived features from source depth maps, to speed up and enhance the quality of estimated decoder-side depth maps [136]. Moreover, the new version of the assistance, Extended Geometry Assistance SEI is adopted for MIV Ed.2 [137]. This version includes the possibility to send the features only for a subset of views, and adds new schemes for feature extraction, apart from a rectangular grid. This could be advantageous in our pDSDE and hpDSDE scenario for a depth estimation speed-up and bitrate reduction.

Furthermore, patch depth estimation (having only a few available views) is significantly more challenging than regular depth estimation from multiple views. Some potential improvements could be considered in this case. First, we could consider using an adapted depth estimator, which would skip the pruned pixels, and make the estimation faster and more accurate. Then, we could use "input depth map assistance", *i.e.,* sending a subset of input depth maps, and using them to refine the depth maps of other views [138] or re-projecting them to recover other depth maps [139], and speed-up the MIV rendering. In addition, monocular depth estimation could be investigated in case there is not enough correspondence among the patches [50].

Moreover, our hpDSDE method needs multiple coding passes in order to find the best threshold for depth patch selection. This is due to the chosen evaluation metric, which is the PSNR between source patch depth and estimated patch depth. However, once selected, there is no need to change the "quality" threshold often, but only occasionally, when the scene characteristic changes. This could also be predicted, for example, using temporal information (TI) analysis [140]. Finally, an improved depth patch selection approach could consider another proxy for rendering quality, *e.g.,* warping.

8.2.2 Pruning for neural image-based rendering

At the time of writing this manuscript, MPEG formed an ad-hoc group for the investigation of implicit neural video representations (INVR) [141]. This group aims to evaluate the novel view rendering possibilities with NeRF-based approaches, especially focused on rendering more complex scenes with non-Lambertian and transparent objects. We believe that our work on input pruning for IBRNet (Chapter 7) perfectly complements these efforts.

However, our work is the first step in the complete immersive video coding pipeline. Further work could assure delivery with MIV, in the following way. The pruning percentage in LeHoPP could be automatically selected based on the given pixel rate limits [20]. Our pruning method could be used instead of TMIV's pruner, and after the pruning step, the process could continue following the TMIV pipeline, by clustering the non-pruned pixels and packing the clusters into patches [24, 89]. Finally, the generated atlases could be compressed and decompressed using a 2D video codec, and the rendering could be done with IBRNet [73].

Neural IBR methods seem to be the future of rendering, but image pruning for neural synthesizers is in its early stage. Pruning should identify the patches which are of an appropriate size, so that they can be easily compressed with 2D codecs. Therefore, we need to cluster the patches during the pruning process. For clustering, we could introduce an additional metric in pixel pruning computation, which would penalize pruned pixels that are isolated. Potentially, other general IBR neural synthesizers could be tested in this scenario [142], or even the fast implicit neural IBR methods [143].

Furthermore, our pruning method, which is supposed to be at the encoder side, introduces additional complexity to the network. This complexity can be reduced, *e.g.,* by moving the operations (such as inpainting) to the GPU, or by using semi-precision rather than single-precision for faster computation (Automatic Mixed Precision in PyTorch) [144]. Additionally, the synthesis performance with pruned input could be improved by the integration of the pruning module in the training.

Finally, temporal consistency is an issue related to neural renderers which are image-based, and do not consider the temporal dimension. Our pruning masks are aggregated during one intra-period to avoid the temporal inconsistencies due to pruning. Moreover, there are multiple works addressing the temporal dimension, *e.g.,* based on joint spatial and temporal information modelling [145] or temporal incoherence modeling and plugging it into image-based framework [146]. Moreover, some works enable the streaming of radiance fields with incremental representation, which speeds up the training and rendering time for novel view synthesis of a 3D video [147, 148].

Appendix A

Study on Depth Estimators for Rendering of Immersive Video

To ensure that all the works conducted during my thesis are included, in this Appendix we present a brief overview of the study[1] led by another PhD student in Orange Labs. The study evaluates two conventional and five learning-based depth estimation techniques for a use case of MIV rendering, for which I personally contributed with advice on the TMIV, IVDE, DERS, and GwcNet software, and with paper review and corrections.

A.1 Summary of the results

As we had the opportunity to see in previous chapters, depth estimation has an essential role in an immersive video coding scheme that relies on DIBR techniques. Most often, depth estimation methods are evaluated for the depth accuracy they achieve. However, in an immersive video system, the main quality indicator is the actual quality of the rendered views. Therefore, the goal of this study is to evaluate the synthesized view quality depending on the depth estimator, where novel views are generated by the TMIV (VWS) renderer, described in Chapter 2.

The study encompasses two conventional and five learning-based depth estimation methods. The tested conventional depth estimators are DERS [55] and IVDE [56], which are described in Chapter 2. Learning-based methods are two stereo methods GwcNet [57] and GA-Net [58], also described in Chapter 2, and additional multi-view stereo approaches: RMVSNet [49], AA-RMVSNet [149], and a neural renderer IBRNet [73] that can be used to compute depth maps. Therefore, this study enlarges the choice of learned depth estimators compared to Chapter 5 by adding the multi-view stereo solutions. All models are tested on perspective and rectified content from the MIV CTC, because almost all models are pre-trained mostly on such data.

On average, synthesis results show that depth maps obtained with the IVDE method yield the highest quality of 38.77 dB, 0.123, and 0.884 in terms of IV-PSNR, LPIPS, and SSIM metric, respectively. These results are very closely followed by GA-Net: 38.36 dB IV-PSNR, 0.125 LPIPS, and 0.884 SSIM; and DERS: 38.34 dB IV-PSNR, 0.124 LPIPS, and 0.882 SSIM. Moreover, when observing the maximum results per sequence, it is noticeable that learned approaches yield better results in more than half of video sequences in terms of IV-PSNR and SSIM, while in terms of LPIPS, the results are balanced.

Subjective analysis is consistent with the one conducted in Chapter 5. It is observable that conventional techniques generate depth maps that are noisier, while

[1] The content of this appendix is based on the work we published in [28].

learned approaches generate depth maps with smoother and cloudy regions. Subsequently, the conventional methods more accurately recover the sharp object edges, while learned approaches struggle with object boundaries and thin structures, due to computational issues with high-resolution data.

A.2 Conclusion

This study has shown that learned-based depth estimators are still not surpassing the capabilities of conventional approaches. This is mostly due to the domain shift issue between natural and computer-generated content, which may arise when using models pre-trained on particular content. Moreover, learning-based methods show computational inefficiency when dealing with high-resolution content, which is the usual scenario for immersive video applications. Therefore, our study highlights the importance of further research for optimizing the learned approaches to deal with wide baselines, high resolutions, and different types of content.

Appendix B

List of Publications

B.1 Publications

1. **M. Milovanović**, F. Henry, M. Cagnazzo and J. Jung, "Patch Decoder-Side Depth Estimation In Mpeg Immersive Video," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toronto, ON, Canada, 2021, pp. 1945-1949.

2. D. Mieloch, P. Garus, **M. Milovanović**, J. Jung, J. Y. Jeong, S. L. Ravi and B. Salahieh, "Overview and Efficiency of Decoder-Side Depth Estimation in MPEG Immersive Video," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 9, pp. 6360-6374, Sept. 2022.

3. S. L. Ravi, **M. Milovanović**, L. Morin and F. Henry, "A Study of Conventional and Learning-Based Depth Estimators for Immersive Video Transmission," in *2022 IEEE 24th International Workshop on Multimedia Signal Processing (MMSP)*, Shanghai, China, 2022, pp. 1-5.

4. **M. Milovanović**, F. Henry and M. Cagnazzo, "Depth Patch Selection for Decoder-Side Depth Estimation in MPEG Immersive Video," in *2022 Picture Coding Symposium (PCS)*, San Jose, CA, USA, 2022, pp. 343-347.

5. P. Garus*, **M. Milovanović***, J. Jung, M. Cagnazzo, "Chapter 12 - MPEG immersive video," in *Immersive Video Technologies*, G. Valenzise, M. Alain, E. Zerman, and C. Ozcinar, Eds., pp. 327-356, Academic Press, 2023.

   *The authors P.G. and M.M. contributed equally to this chapter.

6. **M. Milovanović**, E. Tartaglione, M. Cagnazzo, F. Henry, "Learn how to prune pixels for multi-view neural image-based synthesis," to appear in *2023 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, Brisbane, Australia, 2023, pp. 1-6.

B.2 Patents

1. F. Henry, **M. Milovanović**, "Procédé de segmentation d'une pluralité de données, procédé de codage, procédé de décodage, dispositifs, systèmes et programme d'ordinateur correspondants," French Patent ref. FR2206177, June 2022.

B.3 Standardization contributions

1. **M. Milovanović**, J. Jung, "Improvements of the MV-HEVC+VVS anchor for immersive video," Doc. ISO/IEC JTC1/SC29/WG11 MPEG2019/m49149, Gothenburg, SE, July 2019.

2. **M. Milovanović**, P. Garus, P. Boissonade, J. Jung, "MV-HEVC anchor results for immersive video CTCs and template," Doc. ISO/IEC JTC1/SC29/WG11 MPEG2019/m49093, Gothenburg, SE, July 2019.

3. **M. Milovanović**, J. Jung, P. Boissonade, "TMIV anchor results, analysis, and comparison with MV-HEVC anchor," Doc. ISO/IEC JTC1/SC29/WG11 MPEG2019/m49593, Gothenburg, SE, July 2019.

4. **M. Milovanović**, P. Garus, J. Jung, "Evaluation of MIV with VVC, AV1, AVS3 codecs and with screen content coding tools," Doc. ISO/IEC JTC 1/SC 29/WG 4 m58037, Online, Oct. 2021.

Bibliography

[1] Lik-Hang Lee, Tristan Braud, Pengyuan Zhou, Lin Wang, Dianlei Xu, Zijun Lin, Abhishek Kumar, Carlos Bermejo, and Pan Hui, "All one needs to know about metaverse: A complete survey on technological singularity, virtual ecosystem, and research agenda," 2021.

[2] Martin Alain, Emin Zerman, Cagri Ozcinar, and Giuseppe Valenzise, "Chapter 1 - introduction to immersive video technologies," in *Immersive Video Technologies*, Giuseppe Valenzise, Martin Alain, Emin Zerman, and Cagri Ozcinar, Eds., pp. 3–24. Academic Press, 2023.

[3] Frederic Dufaux, Béatrice Pesquet-Popescu, and Marco Cagnazzo, *Emerging technologies for 3D video: creation, coding, transmission and rendering*, John Wiley & Sons, 2013.

[4] Thibault Lacharme, Mohamed-Chaker Larabi, and Daniel Meneveaux, "Exploration of comfort factors for virtual reality environments," in *Electronic Imaging - Image Quality and System Performance XIX*, San Francisco, United States, Jan. 2022, vol. 34.

[5] Aline Roumy and Thomas Maugey, "Universal lossless coding with random user access: The cost of interactivity," in *2015 IEEE International Conference on Image Processing (ICIP)*, 2015, pp. 1870–1874.

[6] Sam Kavanagh, Andrew Luxton-Reilly, Burkhard Wuensche, and Beryl Plimmer, "A systematic review of virtual reality in education," *Themes in Science and Technology Education*, vol. 10, no. 2, pp. 85–119, 2017.

[7] Philipp Merkle, Aljoscha Smolic, Karsten Muller, and Thomas Wiegand, "Multi-View Video Plus Depth Representation and Coding," in *2007 IEEE International Conference on Image Processing*, San Antonio, TX, USA, Sept. 2007, pp. I – 201–I – 204, IEEE, ISSN: 1522-4880.

[8] Masayuki Tanimoto, "Overview of FTV (free-viewpoint television)," in *2009 IEEE International Conference on Multimedia and Expo*, New York, NY, USA, June 2009, pp. 1552–1553, IEEE.

[9] Bappaditya Ray, Joel Jung, and Mohamed-Chaker Larabi, "On the possibility to achieve 6-dof for 360 video using divergent multi- view content," in *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 211–215.

[10] Christoph Fehn, "Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV," San Jose, CA, May 2004, pp. 93–104.

[11] Sarah Fachada, Daniele Bonatto, Arnaud Schenkel, and Gauthier Lafruit, "Depth image based view synthesis with multiple reference views for virtual reality," in *2018 - 3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, 2018, pp. 1–4.

[12] Navid Mahmoudian Bidgoli, Thomas Maugey, and Aline Roumy, "Correlation model selection for interactive video communication," in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 2184–2188.

[13] Mary-Luc Champel, Thomas Stockhammer, Thierry Fautier, Emmanuel Thomas, and Rob Koenen, "Quality requirements for VR," ISO/IEC JTC 1/SC 29/WG 11 MPEG 2016/m39532, 2016.

[14] Olgierd Stankiewicz, Gauthier Lafruit, and Marek Domański, "Chapter 1 - multiview video: Acquisition, processing, compression, and virtual view rendering," in *Academic Press Library in Signal Processing, Volume 6*, Rama Chellappa and Sergios Theodoridis, Eds., pp. 3–74. Academic Press, 2018.

[15] "Stereoscopic television MPEG-2 multi-view profile," Rep. ITU-R BT.2017, 1998.

[16] Anthony Vetro, Thomas Wiegand, and Gary J. Sullivan, "Overview of the stereo and multiview video coding extensions of the h.264/mpeg-4 avc standard," *Proceedings of the IEEE*, vol. 99, no. 4, pp. 626–642, 2011.

[17] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra, "Overview of the h. 264/avc video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.

[18] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[19] Gerhard Tech, Ying Chen, Karsten Müller, Jens-Rainer Ohm, Anthony Vetro, and Ye-Kui Wang, "Overview of the Multiview and 3D Extensions of High Efficiency Video Coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 1, pp. 35–49, 2016.

[20] Bart Kroon, Vinod Kumar Malamal Vadakital, and Joel Jung, "Recommended pixel rate limits for the CTC for Immersive Video," ISO/IEC JTC 1/SC 29/WG 11 MPEG/m49826, 2019.

[21] Mathias Wien, Jill M. Boyce, Thomas Stockhammer, and Wen-Hsiao Peng, "Standardization Status of Immersive Video Coding," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 9, no. 1, pp. 5–17, Mar. 2019.

[22] Jill M. Boyce, Renaud Doré, Adrian Dziembowski, Julien Fleureau, Joel Jung, Bart Kroon, Basel Salahieh, Vinod Kumar Malamal Vadakital, and Lu Yu, "MPEG Immersive Video Coding Standard," *Proceedings of the IEEE*, vol. 109, no. 9, pp. 1521–1536, 2021.

[23] "Text of ISO/IEC FDIS 23090-12 MPEG Immersive Video," ISO/IEC JTC1/SC29/WG4 MPEG2021/ N00111, July 2021.

[24] Adrian Dziembowski and Basel Salahieh, "Test Model 15 for MPEG Immersive Video," ISO/IEC JTC 1/SC 29/WG 04 N0271, Oct. 2022.

[25] Patrick Garus, Joel Jung, Thomas Maugey, and Christine Guillemot, "Bypassing Depth Maps Transmission For Immersive Video Coding," in *2019 Picture Coding Symposium (PCS)*, 2019, pp. 1–5.

[26] Dawid Mieloch, Patrick Garus, Marta Milovanović, Joël Jung, Jun Young Jeong, Smitha Lingadahalli Ravi, and Basel Salahieh, "Overview and Efficiency of Decoder-Side Depth Estimation in MPEG Immersive Video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 9, pp. 6360–6374, 2022.

[27] Patrick Garus, Marta Milovanović, Joël Jung, and Marco Cagnazzo, "Chapter 12 - MPEG immersive video," in *Immersive Video Technologies*, Giuseppe Valenzise, Martin Alain, Emin Zerman, and Cagri Ozcinar, Eds., pp. 327–356. Academic Press, 2023.

[28] Smitha Lingadahalli Ravi, Marta Milovanović, Luce Morin, and Félix Henry, "A study of conventional and learning-based depth estimators for immersive video transmission," in *2022 IEEE 24th International Workshop on Multimedia Signal Processing (MMSP)*, 2022, pp. 1–5.

[29] Marta Milovanović, Félix Henry, Marco Cagnazzo, and Joël Jung, "Patch Decoder-Side Depth Estimation In Mpeg Immersive Video," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 1945–1949.

[30] Marta Milovanović, Félix Henry, and Marco Cagnazzo, "Depth patch selection for decoder-side depth estimation in mpeg immersive video," in *2022 Picture Coding Symposium (PCS)*, 2022, pp. 343–347.

[31] Marta Milovanović, Enzo Tartaglione, Marco Cagnazzo, and Félix Henry, "Learn how to prune pixels for multi-view neural image-based synthesis," 2023.

[32] Brian Rogers and Maureen Graham, "Motion parallax as an independent cue for depth perception," *Perception*, vol. 8, no. 2, pp. 125–134, 1979.

[33] Basel Salahieh, Binoy Marvar, Michael Mefenza Nentedem, Avinash Kumar, Vladan Popovic, Kalpana Seshadrinathan, Oscar Nestares, and Jill Boyce, "Kermit test sequence for Windowed 6DoF Activities," ISO/IEC JTC 1/SC 29/WG 11 MPEG2018/M43748, 2018.

[34] Eugene d'Eon, Bob Harrison, Taos Myers, and Philip A. Chou, "8i Voxelized Full Bodies – A Voxelized Point Cloud Dataset," ISO/IEC JTC 1/SC 29/WG 11 m40059, Jan. 2017.

[35] Giuseppe Valenzise, Maurice Quach, Dong Tian, Jiahao Pang, and Frédéric Dufaux, "Chapter 13 - Point cloud compression," in *Immersive Video Technologies*, Giuseppe Valenzise, Martin Alain, Emin Zerman, and Cagri Ozcinar, Eds., pp. 357–385. Academic Press, 2023.

[36] Mathias Wien, ," in *High Efficiency Video Coding - Coding Tools and Specification*. Springer Berlin, Heidelberg, 2015.

[37] Tung Nguyen, Xiaozhong Xu, Felix Henry, Ru-Ling Liao, Mohammed Golam Sarwer, Marta Karczewicz, Yung-Hsuan Chao, Jizheng Xu, Shan Liu, Detlev Marpe, and Gary J. Sullivan, "Overview of the screen content support in vvc: Applications, coding tools, and performance," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 3801–3817, 2021.

[38] Mohsen Abdoli, Felix Henry, Patrice Brault, Frédéric Dufaux, Pierre Duhamel, and Pierrick Philippe, "Intra block-dpcm with layer separation of screen content in vvc," in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 3162–3166.

[39] Jingning Han, Bohan Li, Debargha Mukherjee, Ching-Han Chiang, Adrian Grange, Cheng Chen, Hui Su, Sarah Parker, Sai Deng, Urvang Joshi, Yue Chen, Yunqing Wang, Paul Wilkins, Yaowu Xu, and James Bankoski, "A Technical Overview of AV1," *Proceedings of the IEEE*, vol. 109, no. 9, pp. 1435–1462, 2021.

[40] "Alliance for Open Media," `https://aomedia.org/`, [Online; accessed 30-December-2021].

[41] Jiaqi Zhang, Chuanmin Jia, Meng Lei, Shanshe Wang, Siwei Ma, and Wen Gao, "Recent Development of AVS Video Coding Standard: AVS3," in *2019 Picture Coding Symposium (PCS)*, 2019, pp. 1–5.

[42] "Audio and Video Coding Standard Workgroup of China," `http://www.avs.org.cn/english/`, [Online; accessed 30-December-2021].

[43] Tung Nguyen and Detlev Marpe, "Future video coding technologies: A performance evaluation of av1, jem, vp9, and hm," in *2018 picture coding symposium (PCS)*. IEEE, 2018, pp. 31–35.

[44] Gerhard Tech, Ying Chen, Karsten Muller, Jens-Rainer Ohm, Anthony Vetro, and Ye-Kui 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.

[45] D Graziosi, O Nakagami, S Kuma, A Zaghetto, T Suzuki, and A Tabatabai, "An overview of ongoing point cloud compression standardization activities: Video-based (v-pcc) and geometry-based (g-pcc)," *APSIPA Transactions on Signal and Information Processing*, vol. 9, pp. e13, 2020.

[46] S Burak Gokturk, Hakan Yalcin, and Cyrus Bamji, "A time-of-flight depth sensor-system description, issues and solutions," in *2004 conference on computer vision and pattern recognition workshop*. IEEE, 2004, pp. 35–35.

[47] Sen Xiang, Li Yu, Qiong Liu, and Zixiang Xiong, "A gradient-based approach for interference cancelation in systems with multiple kinect cameras," in *2013 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2013, pp. 13–16.

[48] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3828–3838.

[49] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan, "Mvsnet: Depth inference for unstructured multi-view stereo," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 767–783.

[50] Diana Wofk, Fangchang Ma, Tien-Ju Yang, Sertac Karaman, and Vivienne Sze, "Fastdepth: Fast monocular depth estimation on embedded systems," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 6101–6108.

[51] Andreas Geiger, Philip Lenz, and Raquel Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[52] Jingyu Yang, Zhongyu Jiang, Xinchen Ye, and Kun Li, "Depth super-resolution with color guidance: a review," *RGB-D Image Analysis and Processing*, pp. 51–65, 2019.

[53] Yan Li, Qiong Wang, Lu Zhang, and Gauthier Lafruit, "A lightweight depth estimation network for wide-baseline light fields," *IEEE Transactions on Image Processing*, vol. 30, pp. 2288–2300, 2021.

[54] Matteo Poggi, Fabio Tosi, Konstantinos Batsos, Philippos Mordohai, and Stefano Mattoccia, "On the synergies between machine learning and binocular stereo for depth estimation from images: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5314–5334, 2021.

[55] Takanori Senoh, Nobuji Tetsutani, Hiroshi Yasuda, and Mehrdad Teratani, "Revised Proposed Depth Estimation Reference Software (pDERS8.1)," ISO/IEC JTC 1/SC 29/WG 11 MPEG 2018/M45265, Jan. 2019.

[56] Dawid Mieloch, "Manual of Immersive Video Depth Estimation 3," ISO/IEC JTC 1/SC 29/WG 4 MPEG/N0058, Jan. 2021.

[57] Xiaoyang Guo, Kai Yang, Wukui Yang, Xiaogang Wang, and Hongsheng Li, "Group-wise correlation stereo network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3273–3282.

[58] Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr, "Ga-net: Guided aggregation net for end-to-end stereo matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 185–194.

[59] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge books online. Cambridge University Press, 2003.

[60] Margarita Chli, Roland Siegwart, and Nick Lawrance, "Perception II - Fundamentals of Computer Vision," `https://www.edx.org/course/autonomous-mobile-robots/`, 2023, [Online; accessed 15-May-2023].

[61] Krzysztof Wegner, Olgierd Stankiewicz, Tomasz Grajek, and Marek Domański, "Depth Map Formats Used Within MPEG 3D Technologies," ISO/IEC JTC 1/SC 29/WG 11 MPEG 2017/N16730, Jan. 2017.

[62] Takanori Senoh, Nobuji Tetsutani, and Hiroshi Yasuda, "Depth Estimation and View Synthesis for Immersive Media," in *2018 International Conference on 3D Immersion (IC3D)*, Brussels, Belgium, Dec. 2018, pp. 1–8, IEEE.

[63] Segolene Rogge, Daniele Bonatto, Jaime Sancho, Ruben Salvador, Eduardo Juarez, Adrian Munteanu, and Gauthier Lafruit, "MPEG-I Depth Estimation Reference Software," in *2019 International Conference on 3D Immersion (IC3D)*, Brussels, Belgium, Dec. 2019, pp. 1–6, IEEE.

[64] Dawid Mieloch, Olgierd Stankiewicz, and Marek Domanski, "Depth Map Estimation for Free-Viewpoint Television and Virtual Navigation," *IEEE Access*, vol. 8, pp. 5760–5776, 2020.

[65] Yuri Boykov and Vladimir Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 9, pp. 1124–1137, 2004.

[66] Andrew Blake, Pushmeet Kohli, and Carsten Rother, *Markov random fields for vision and image processing*, MIT press, 2011.

[67] Radhakrishna Achanta and Sabine Susstrunk, "Superpixels and polygons using simple non-iterative clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4651–4660.

[68] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4040–4048.

[69] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[70] "Middlebury Stereo Evaluation," `https://vision.middlebury.edu/stereo/eval3/`, 2023, [Online; accessed 15-May-2023].

[71] "The KITTI Vision Benchmark Suite," `https://www.cvlibs.net/datasets/kitti/eval_stereo.php/`, 2023, [Online; accessed 15-May-2023].

[72] Adriano Q de Oliveira, Thiago LT da Silveira, Marcelo Walter, and Cláudio R Jung, "On the performance of dibr methods when using depth maps from state-of-the-art stereo matching algorithms," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 2272–2276.

[73] Harry Shum and Sing Bing Kang, "Review of image-based rendering techniques," in *Visual Communications and Image Processing 2000*, King N. Ngan, Thomas Sikora, and Ming-Ting Sun, Eds. International Society for Optics and Photonics, 2000, vol. 4067, pp. 2 – 13, SPIE.

[74] Beerend Ceulemans, Shao-Ping Lu, Gauthier Lafruit, and Adrian Munteanu, "Robust multiview synthesis for wide-baseline camera arrays," *IEEE Transactions on Multimedia*, vol. 20, no. 9, pp. 2235–2248, 2018.

[75] Sarah Fachada, Daniele Bonatto, Mehrdad Teratani, and Gauthier Lafruit, "View synthesis tool for vr immersive video," 2022.

[76] Daniele Bonatto, Sarah Fachada, Ségolène Rogge, Adrian Munteanu, and Gauthier Lafruit, "Real-time depth video-based rendering for 6-dof hmd navigation and light field displays," *IEEE access*, vol. 9, pp. 146868–146887, 2021.

[77] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar, "Local light field fusion: Practical view synthesis with prescriptive sampling guidelines," *ACM Transactions on Graphics (TOG)*, 2019.

[78] Eric Penner and Li Zhang, "Soft 3d reconstruction for view synthesis," vol. 36, no. 6, 2017.

[79] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng, "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," in *European conference on computer vision*. Aug 2020, pp. 405–421, Springer.

[80] "Call for Proposals on 3DoF+ Visual," ISO/IEC JTC 1/SC 29/WG 11 MPEG/N18145, Jan. 2019.

[81] Vinod Kumar Malamal Vadakital, Adrian Dziembowski, Gauthier Lafruit, Franck Thudor, Gwangsoon Lee, and Patrice Rondao Alface, "The mpeg immersive video standard—current status and future outlook," *IEEE MultiMedia*, vol. 29, no. 3, pp. 101–111, 2022.

[82] "MPEG Immersive Video (MIV)," `https://mpeg-miv.org/`, 2023, [Online; accessed 15-May-2023].

[83] Tomás Völker, Guillaume Boisson, and Bertrand Chupeau, "Learning light field synthesis with multi-plane images: scene encoding as a recurrent segmentation task," in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 633–637.

[84] Patrick Garus, Felix Henry, Joël Jung, Thomas Maugey, and Christine Guillemot, "Immersive video coding: Should geometry information be transmitted as depth maps?," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 5, pp. 3250–3264, 2021.

[85] Patrick Garus, Felix Henry, Thomas Maugey, and Christine Guillemot, "Motion compensation-based low-complexity decoder side depth estimation for mpeg immersive video," in *2022 IEEE 24th International Workshop on Multimedia Signal Processing (MMSP)*. IEEE, 2022, pp. 1–6.

[86] P.J. Burt, "Moment images, polynomial fit filters. and the problem of surface interpolation," in *Proceedings CVPR '88: The Computer Society Conference on Computer Vision and Pattern Recognition*, 1988, pp. 144–152.

[87] "Partitioning Around Medoids (Program PAM)," in *Wiley Series in Probability and Statistics*. John Wiley & Sons.

[88] Hong-Chang Shin, Jun-Young Jeong, Gwangsoon Lee, Muhammad Umer Kakli, Junyoung Yun, and Jeongil Seo, "Enhanced pruning algorithm for improving visual quality in MPEG immersive video," *ETRI Journal*, vol. 44, no. 1, pp. 73–84, 2022.

[89] Jukka Jylänki, "A Thousand Ways to Pack the Bin - A Practical Approach to Two-Dimensional Rectangle Bin Packing," `http://pds25.egloos.com/pds/201504/21/98/RectangleBinPack.pdf/`, 2010, [Online; accessed 07-May-2023].

[90] Danillo B. Graziosi and Bart Kroon, "Video-based coding of volumetric data," in *2020 IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 2706–2710.

[91] "Text of ISO/IEC DIS 23090-5 Visual Volumetric Video-based Coding and Video-based Point Cloud Compression 2nd Edition," ISO/IEC JTC1/SC29/WG7 MPEG2021/ N00188, July 2021.

[92] Basel Salahieh, Bart Kroon, Joel Jung, and Marek Domański, "Test Model 4 for Immersive Video," ISO/IEC JTC 1/SC 29/WG 11 N19002, Feb. 2020.

[93] Basel Salahieh, Bart Kroon, Joel Jung, Bart Kroon, and Adrian Dziembowski, "Test Model 7 for MPEG Immersive Video," ISO/IEC JTC 1/SC 29/WG 4 N0005, Oct. 2020.

[94] Joel Jung and Bart Kroon, "Common Test Conditions for MPEG Immersive Video," ISO/IEC JTC 1/SC 29/WG 04 N0113, 2021.

[95] "The Fraunhofer Versatile Video Encoder (VVenC)," `https://github.com/fraunhoferhhi/vvenc/`, 2019, [Online; accessed 30-October-2021].

[96] Adam Wieckowski, Jens Brandenburg, Tobias Hinz, Christian Bartnik, Valeri George, Gabriel Hege, Christian Helmrich, Anastasia Henkel, Christian Lehmann, Christian Stoffers, Ivan Zupancic, Benjamin Bross, and Detlev Marpe, "Vvenc: An Open And Optimized Vvc Encoder Implementation," in *2021 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*, 2021, pp. 1–2.

[97] "Versatile Video Coding (VVC)," Doc. ITU-T Rec. H.266, ISO/IEC 23090-3, Feb. 2021.

[98] Joel Jung, Gilles Teniou, Xiang Li, and Shan Liu, "Moving to VVC for MIV CTC anchor," ISO/IEC JTC 1/SC 29/WG 04 M55823, Jan. 2021.

[99] "The Fraunhofer Versatile Video Decoder (VVdeC)," `https://github.com/fraunhoferhhi/vvdec/`, 2018, [Online; accessed 30-October-2021].

[100] Bart Kroon, "Comments on the common test conditions," ISO/IEC JTC 1/SC 29/WG 11 M54362, June 2020.

[101] Joel Jung, Bart Kroon, and Jill Boyce, "Common Test Conditions for Immersive Video," ISO/IEC JTC 1/SC 29/WG 11 N18997, Jan. 2020.

[102] "Hevc hm reference software," `https://vcgit.hhi.fraunhofer.de/jvet/HM/`, 2023, [Online; accessed 15-May-2023].

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

[104] S. Pateux and J. Jung, "An excel add-in for computing Bjontegaard metric and its evolution," ITU-T SG16 Q 6, Doc. VCEG-AE07, Jan. 2007.

[105] Adrian Dziembowski and Marek Domanski, "Objective quality metric for immersive video," ISO/IEC JTC 1/SC 29/WG 11 M48093, July 2019.

[106] Adrian Dziembowski, Dawid Mieloch, Jakub Stankowski, and Adam Grzelka, "Iv-psnr—the objective quality metric for immersive video applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 11, pp. 7575–7591, 2022.

[107] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[108] Z. Wang, E.P. Simoncelli, and A.C. Bovik, "Multiscale structural similarity for image quality assessment," in *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, Pacific Grove, CA, USA, 2003, pp. 1398–1402, IEEE.

[109] Zhi Li, Anne Aaron, Ioannis Katsavounidis, Anush Moorthy, and Megha Manohara, "Toward a practical perceptual video quality metric," *The Netflix Tech Blog*, 2016.

[110] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 586–595.

[111] Danillo B. Graziosi and Bart Kroon, "Video-Based Coding Of Volumetric Data," in *2020 IEEE International Conference on Image Processing (ICIP)*, Abu Dhabi, United Arab Emirates, Oct. 2020, pp. 2706–2710, IEEE.

[112] Guido Meardi, Simone Ferrara, Lorenzo Ciccarelli, Guendalina Cobianchi, Stergios Poularakis, Florian Maurer, Stefano Battista, and Ahmad Byagowi, "Mpeg-5 part 2: Low complexity enhancement video coding (lcevc): Overview and performance evaluation," *Applications of Digital Image Processing XLIII*, vol. 11510, pp. 238–257, 2020.

[113] Jarosław Samelak, Adrian Dziembowski, Dawid Mieloch, Marek Domański, and Maciej Wawrzyniak, "Efficient Immersive Video Compression using Screen Content Coding," 05 2021, pp. 197–206.

[114] Xiaozhong Xu and Shan Liu, "Overview of screen content coding in recently developed video coding standards," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 2, pp. 839–852, 2021.

[115] Jarosław Samelak, Adrian Dziembowski, Dawid Mieloch, Marek Domański, and Maciej Wawrzyniak, "Efficient immersive video compression using screen content coding," *29. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, pp. 197–206, 2021.

[116] Joel Jung, Gilles Teniou, Xiang Li, and Shan Liu, "Moving to VVC for MIV CTC anchor," ISO/IEC JTC 1/SC 29/WG 04 MPEG/m55823, Jan. 2021.

[117] Mohsen Abdoli, Félix Henry, Patrice Brault, Pierre Duhamel, and Frédéric Dufaux, "Short-Distance Intra Prediction of Screen Content in Versatile Video Coding (VVC)," *IEEE Signal Processing Letters*, vol. 25, no. 11, pp. 1690–1694, 2018.

[118] Marta Milovanović, "View synthesis and video compression for next generation virtual reality applications," Tech. Rep., Télécom Paris ; Université Paris-Saclay ; Orange Labs, Sept. 2019, (Unpublished internship report).

[119] Mary-Luc Champel, Thomas Stockhammer, Thierry Fautier, Emmanuel Thomas, and Rob Koenen, "Quality requirements for vr," in *116th MPEG meeting of ISO/IEC JTC1/SC29/WG11, MPEG*, 2016, vol. 116, p. m39532.

[120] Jiangbo Lu, Qiong Yang, and Gauthier Lafruit, "Interpolation error as a quality metric for stereo: Robust, or not?," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2009, pp. 977–980.

[121] Joel Jung and Bart Kroon, "Common Test Conditions for MPEG Immersive Video," ISO/IEC JTC 1/SC 29/WG 04 0051, Oct. 2020.

[122] "Test Model 10 for MPEG Immersive Video," ISO/IEC JTC1/SC29/WG4 MPEG2021/ N0112, July 2021.

[123] Jong-Beom Jeong, Soonbin Lee, Dongmin Jang, and Eun-Seok Ryu, "Towards 3DoF+ 360 Video Streaming System for Immersive Media," *IEEE Access*, vol. 7, pp. 136399–136408, 2019.

[124] Hong-Chang Shin, Jun-Young Jeong, Gwangsoon Lee, Muhammad Umer Kakli, Junyoung Yun, and Jeongil Seo, "Enhanced pruning algorithm for improving visual quality in MPEG immersive video," *ETRI Journal*, vol. 44, no. 1, pp. 73–84, 2022.

[125] Patrick Garus, Félix Henry, Joel Jung, Thomas Maugey, and Christine Guillemot, "Immersive video coding: Should geometry information be transmitted as depth maps?," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 5, pp. 3250–3264, 2022.

[126] Adrian Dziembowski, Dawid Mieloch, Jakub Stankowski, and Adam Grzelka, "IV-PSNR—The Objective Quality Metric for Immersive Video Applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 11, pp. 7575–7591, 2022.

[127] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16000–16009.

[128] Jihoon Tack, Subin Kim, Sihyun Yu, Jaeho Lee, Jinwoo Shin, and Jonathan Richard Schwarz, "Efficient meta-learning via error-based context pruning for implicit neural representations," *arXiv preprint arXiv:2302.00617*, 2023.

[129] Enzo Tartaglione, Andrea Bragagnolo, Attilio Fiandrotti, and Marco Grangetto, "Loss-based sensitivity regularization: towards deep sparse neural networks," *Neural Networks*, vol. 146, pp. 230–237, 2022.

[130] Chenxi Lola Deng and Enzo Tartaglione, "Compressing explicit voxel grid representations: Fast nerfs become also small," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, January 2023, pp. 1236–1245.

[131] Alexandru Telea, "An image inpainting technique based on the fast marching method," *Journal of Graphics Tools*, vol. 9, 2004.

[132] Dawid Mieloch, Adrian Dziembowski, Jun Young Jeong, and Gwangsoon Lee, "On the future of decoder-side depth estimation in mpeg immersive video coding," 2023.

[133] Adrian Dziembowski, Dawid Mieloch, Jun Young Jeong, and Gwangsoon Lee, "MIV decoder-side depth estimation profile," Oct. 2022.

[134] Adrian Dziembowski, Dawid Mieloch, Marek Domański, Lee Gwangsoon, and Jun Young Jeong, "Spatiotemporal redundancy removal in immersive video coding," 2022.

[135] Dawid Mieloch, Adrian Dziembowski, and Marek Domański, "Depth map refinement for immersive video," *IEEE Access*, vol. 9, pp. 10778–10788, 2021.

[136] Gordon Clare, Patrick Garus, and Felix Henry, "[MIV] Geometry Assistance SEI message," Apr. 2021.

[137] Adrian Dziembowski, Dawid Mieloch, Jun Young Jeong, and Gwangsoon Lee, "[MIV] Extended geometry assistance SEI," 2022.

[138] Dominika Klóska, Dawid Mieloch, Adrian Dziembowski, Marek Domański, Lee Gwangsoon, and Jun Young Jeong, "Decoder-side depth estimation with input depth assistance," Oct. 2021.

[139] Dawid Mieloch, Adrian Dziembowski, Blazej Szydelko, Dominika Klóska, Jun Young Jeong, and Gwangsoon Lee, "[MIV] Extended geometry assistance SEI," Apr. 2022.

[140] "Spatial Information (SI) and Temporal Information (TI) calculation," `https://github.com/NabajeetBarman/SI-TI/`, 2019, [Online; accessed 15-May-2023].

[141] Gauthier Lafruit, "BoG report on Neural implicit video representation," ISO/IEC JTC 1/SC 29/WG 04 m60589, 2022.

[142] Yuan Liu, Sida Peng, Lingjie Liu, Qianqian Wang, Peng Wang, Christian Theobalt, Xiaowei Zhou, and Wenping Wang, "Neural rays for occlusion-aware image-based rendering," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 7814–7823.

[143] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, no. 4, jul 2022.

[144] "Automatic mixed precision package - torch.amp," `https://pytorch.org/docs/stable/amp.html/`, 2023, [Online; accessed 15-May-2023].

[145] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, and Zhaoyang Lv, "Neural 3d video synthesis from multi-view video," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 5511–5521.

[146] Smitha Lingadahalli Ravi, Félix Henry, Luce Morin, and Matthieu Gendrin, "Exploring temporal consistency in image-based rendering for immersive video transmission," in *2022 10th European Workshop on Visual Information Processing (EUVIP)*, 2022, pp. 1–6.

[147] Lingzhi Li, Zhen Shen, zhongshu wang, Li Shen, and Ping Tan, "Streaming radiance fields for 3d video synthesis," in *Advances in Neural Information Processing Systems*, Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, Eds., 2022.

[148] Shengze Wang, Alexey Supikov, Joshua Ratcliff, Henry Fuchs, and Ronald Azuma, "Inv: Towards streaming incremental neural videos," 2023.

[149] Zizhuang Wei, Qingtian Zhu, Chen Min, Yisong Chen, and Guoping Wang, "Aa-rmvsnet: Adaptive aggregation recurrent multi-view stereo network," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6187–6196.

ECOLE
DOCTORALE

Titre : Pruning et compression de contenus multi-vues pour le codage vidéo immersif

Mots clés : codage vidéo immersif, MPEG Immersive Video, traitement vidéo, synthèse, carte de profondeur, compression vidéo

Résumé : Cette thèse aborde le problème de la compression efficace de contenus vidéo immersifs, représentés avec le format Multiview Video plus Depth (MVD). Le standard du Moving Picture Experts Group (MPEG) pour la transmission des données MVD est appelé MPEG Immersive Video (MIV), qui utilise des codecs vidéo 2D compresser les informations de texture et de profondeur de la source. Par rapport au codage vidéo traditionnel, le codage vidéo immersif est complexe et limité non seulement par le compromis entre le débit binaire et la qualité, mais aussi par le débit de pixels. C'est pourquoi la MIV utilise le pruning pour réduire le débit de pixels et les corrélations entre les vues et crée une mosaïque de morceaux d'images (patches). L'estimation de la profondeur côté décodeur (DSDE) est apparue comme une approche alternative pour améliorer le système vidéo immersif en évitant la transmission de cartes de profondeur et en déplaçant le processus d'estimation de la profondeur du côté du décodeur. DSDE a été étudiée dans le cas de nombreuses vues entièrement transmises (sans pruning). Dans cette thèse, nous démontrons les avancées possibles en matière de codage vidéo immersif, en mettant l'accent sur le pruning du contenu de source. Nous allons au-delà du DSDE et examinons l'effet distinct de la restauration de la profondeur au niveau du patch du côté du décodeur. Nous proposons deux approches pour intégrer la DSDE sur le contenu traité avec le pruning du MIV. La première approche exclut un sous-ensemble de cartes de profondeur de la transmission, et la seconde approche utilise la qualité des patchs de profondeur estimés du côté de l'encodeur pour distinguer ceux qui doivent être transmis de ceux qui peuvent être récupérés du côté du décodeur. Nos expériences montrent un gain de 4.63 BD-rate pour Y-PSNR en moyenne. En outre, nous étudions également l'utilisation de techniques neuronales de synthèse basées sur l'image (IBR) pour améliorer la qualité de la synthèse de nouvelles vues et nous montrons que la synthèse neuronale elle-même fournit les informations nécessaires au pruning du contenu. Nos résultats montrent un bon compromis entre le taux de pixels et la qualité de la synthèse, permettant d'améliorer la synthèse visuelle de 3.6 dB en moyenne.

Title : Pruning and Compression of Multi-view Content for Immersive Video Coding

Keywords : immersive video coding, MPEG Immersive Video, video processing, rendering, depth map, video compression

Abstract : This thesis addresses the problem of efficient compression of immersive video content, represented with Multiview Video plus Depth (MVD) format. The Moving Picture Experts Group (MPEG) standard for the transmission of MVD data is called MPEG Immersive Video (MIV), which utilizes 2D video codecs to compress the source texture and depth information. Compared to traditional video coding, immersive video coding is more complex and constrained not only by trade-off between bitrate and quality, but also by the pixel rate. Because of that, MIV uses pruning to reduce the pixel rate and inter-view correlations and creates a mosaic of image pieces (patches). Decoder-side depth estimation (DSDE) has emerged as an alternative approach to improve the immersive video system by avoiding the transmission of depth maps and moving the depth estimation process to the decoder side. DSDE has been studied for the case of numerous fully transmitted views (without pruning). In this thesis, we demonstrate possible advances in immersive video coding, emphasized on pruning the input content. We go beyond DSDE and examine the distinct effect of patch-level depth restoration at the decoder side. We propose two approaches to incorporate decoder-side depth estimation (DSDE) on content pruned with MIV. The first approach excludes a subset of depth maps from the transmission, and the second approach uses the quality of depth patches estimated at the encoder side to distinguish between those that need to be transmitted and those that can be recovered at the decoder side. Our experiments show 4.63 BD-rate gain for Y-PSNR on average. Furthermore, we also explore the use of neural image-based rendering (IBR) techniques to enhance the quality of novel view synthesis and show that neural synthesis itself provides the information needed to prune the content. Our results show a good trade-off between pixel rate and synthesis quality, achieving the view synthesis improvements of 3.6 dB on average.

Institut Polytechnique de Paris
91120 Palaiseau, France