Light Field Video Processing and Streaming Using Applied AI

Xinjue Hu

Thesis submitted to the University of Ottawa
in partial Fulfillment of the requirements for the
Doctor of Philosophy degree in Electrical and Computer Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

As a new form of volumetric media, a Light Field (LF) can provide users with a true 6 Degrees-Of-Freedom (DOF) immersive experience, because LF captures the scene with photo-realism, including aperture-limited changes in viewpoint. Nevertheless, the larger size and higher dimension of LF data bring greater challenges to processing and transmission. The main focus of this study is the application of the applied Artificial Intelligence (AI) method to the transmission and processing of LF data, thereby alleviating the performance bottleneck in existing methods.

Uncompressed LF data are too large for network transmission, which is why LF compression has become an important research topic. A new LF compression algorithm based on Graph Neural Network (GNN) is proposed in this work. It can use the graph network model to fit the similarity between the LF viewpoints, so that only the data of a few essential anchor viewpoints need to be transmitted after compression, and a complete LF matrix can be reconstructed according to the graph model at the decoding end. This method also solves the problem of weak generalization of the LF reconstruction algorithm when dealing with high-frequency components through the design of two-layer compression structure. Compared with existing compression methods, a higher compression ratio and better quality can be achieved using this algorithm. Furthermore, to improve the adaptability of the real-time requirements of different LF applications and robustness requirements in unreliable network environments, an adaptive LF video transmission scheme based on Multiple Description Coding (MDC) is proposed. It can divide the LF matrix into LF descriptions at different levels of downsampling ratios, and optimize the scheduling of the descriptions transmission queue, which can ensure that it can adaptively adjust the design of basic GNN unit so that the proposed method can adapt more flexibly to the real-time changes of user viewpoint requests, so as to save unnecessary viewpoint transmission overhead to the greatest extent, and minimize the adverse impact of network packet loss and network status fluctuations on LF transmission services.

For LF processing, depth estimation has been a very hot topic in recent years. To achieve a good balance between the performance of both narrow- or wide-baseline LF data, a novel optical-flow-based LF depth estimation scheme, which uses a convolutional neural network (CNN) to predict the patch matrix after optical flow offset, is proposed. After the optical-flow-assisted offset, the disparity between patches is processed to a unified numerical range, which can effectively solve the overfitting problem of the LF depth estimation network caused by the uneven distribution of the baseline range of LF samples. Experimental results show that the proposed uniform-patch-based estimation mechanism has good generalization on LF data of different baselines and is compatible with various

existing narrow-baseline LF depth estimation algorithms. Finally, since LF processing places high requirements on both the computing and caching capabilities of the infrastructure, a framework that combines Multi-access Edge Computing (MEC) technology with LF applications is proposed in this thesis. In this study, the problem is transformed by the Lyapunov optimization, and an optimized search algorithm based on the Markov approximation method is designed, which can adaptively schedule and adjust the task offloading strategy and resource allocation scheme, so as to provide users with the best service experience in the LF viewpoint interpolation task. Numerical results demonstrate that this edge-based framework can achieve a dynamic balance between energy and caching consumption while meeting the low latency requirements of LF applications.

Acknowledgements

First, I would like to give my heartfelt thanks to all the people who have ever helped me with this thesis.

My sincere and hearty thanks and appreciation go first to my supervisor at the University of Ottawa, Prof. Shervin Shirmohammadi, whose suggestions and encouragement have given me much insight into my studies and research. It has been a great privilege and joy to study under his guidance and supervision. Furthermore, it has been my honor to benefit from his personality and diligence, which I will treasure my whole life. My gratitude to him knows no bounds.

I am also extremely grateful to all my friends and colleagues who have kindly provided me with assistance and companionship in the preparation of this thesis.

In particular, I am also very grateful to my supervisor at my home university, Lin Zhang, who gave me substantial support and guidance during my studies.

Finally, many thanks go to my family and my girlfriend for their unfailing love and unwavering support.

Table of Contents

List of Tables

List of Figures

Acronyms

Acronym	Meaning
AI	Artificial Intelligence
ARQ	Automatic Repeat reQuest
AVC	Advanced Video Coding
BP	Bad Pixel
CDF	Cumulative Distribution Function
CNN	Convolutional Neural Network
DASH	Dynamic Adaptive Streaming over HTTP
DGL	Deep Graph Library
DOF	Degrees Of Freedom
EPI	Epipolar Plane Image
ERC	Error Resilient Coding
FEC	Forward Error Correction
FFT	Fast Fourier Transform
EN	Edge Node
FOV	Field OF Views
FPS	Frames Per Second
GBL	Graph-Based Learning
GNN	Graph Neural Networks
HEVC	High Efficiency Video Coding
IPTV	Internet Protocol TeleVision
IoT	Internet of Things
JPEG	Joint Photographic Experts Group
KKT	Karush-Kuhn-Tucker
LF	Light Field
MAE	Mean Absolute Error
Mbps	Megabits per second
MDC	Multiple Description Coding
MEC	Multi-access Edge Computing
MINLP	Mixed Integer Non-Linear Programming
MPI	Multi-Plane Image
MSE	Mean Square Error

Acronym	Meaning
OTT	Over The Top
PSNR	Peak Signal to Noise Ratio
QUIC	Quick UDP Internet Connections
SAI	Sub-Aperture Image
SFFT	Sparse Fast Fourier Transform
SSIM	Structural SIMilarity index
UE	User's Equipment
VP	ViewPoint

Chapter 1

Introduction

1.1 Motivation

A light field (LF) is a vector that describes the light rays flowing in every direction at each point in space. LF video can capture and display more structure and texture details of the scene compared to conventional 2D/3D/multiview video. In addition, an LF allows a viewer to change the pitch, yaw, and roll of the viewing angle as well as the x, y and z spatial coordinates of the viewing direction, offering a 6 DOF (Degrees Of Freedom) immersive experience. Due to these characteristics, LFs are being proposed for next generation, highly interactive telepresence and immersive systems. For example, in [4], an LF is used to build a highly advanced 3D telemedicine system that provides a life-like teleconsultation experience with a quality of experience far beyond that of conventional telemedicine systems. Another example is TeleHuman2 [5], which is able to convey motion parallax and stereoscopy without a head-worn apparatus, providing full-body 3D video of remote participants using a human-sized cylindrical LF display. An LF has also been used for displaying panoramic still photography in virtual reality [6].

While an LF can provide more scene information than competing technologies, it also introduces new challenges for data compression and transmission. Due to the large amount of information, transmitting LF is a heavy burden for the network. For example, for an LF video with 5×5 viewpoints, each with a resolution of 1024×1024, compression at 30 FPS (Frames Per Second) requires an estimated bitrate that exceeds 100 Mbps (Megabits per second) with H.264/AVC (Advanced Video Coding) and 70 Mbps with H.265/HEVC (High Efficiency Video Coding). As such, there is a need to design effective compression

schemes specifically for LF video. The first goal of this study is to determine how to use limited network resources for efficient LF data transmission.

Although a more efficient and generalized LF compression scheme can effectively reduce the burden of LF services on the network, it still cannot fully meet the needs of LF video transmission for changeable network situations. The second goal of this study is to design a dynamic separable LF coding scheme that is capable of adapting to the real-time requirements of different LF applications and best-effort network conditions causing packet loss.

Depth estimation for LF processing is an important component in the LF image processing pipeline because it directly affects the accuracy of the generated 3D model, which in turn plays a very important role in 3D reconstruction. However, the input baseline of the SAI (SubAperture Image) matrix, which is defined as the physical spatial distance between two adjacent lenses, limits the depth estimation algorithm's performance on different LF acquisition devices. Most of the proposed algorithms are designed for narrow-baseline LF acquired by lenslet cameras and do not perform well for wide-baseline LF acquired by a camera matrix. Conversely, the few algorithms that have been proposed for wide-baseline LF suffer from either requiring prior information in the range of disparity or can only scale the range of disparity at the expense of granularity. A robust and general method is needed to deal with the variety of LF data sources, which is the third goal of this study.

Higher-complexity compression schemes and processing algorithms all place extremely high computing power and energy consumption requirements on the LF deployment platform. In the post-Moore's Law era, it is foreseeable that the development of hardware device capabilities may not fully keep up with the needs of LF technology, and it will also bring high personal device upgrade costs to users. Multiaccess edge computing (MEC) technology, which allows users to lease public computing resources as needed, is one of the most promising ways to address this difficulty. The last goal of this study is to explore the feasibility of combining LF applications with an MEC architecture to realize an LF service pattern that is more suitable for large-scale and low-cost deployment.

1.2 Research Challenges

In this section, the challenges addressed in this study are presented for both LF streaming and processing.

1.2.1 Challenge of LF Compression

The viewpoints of LF are highly similar; however, the most important scene structure information is hidden in the very small disparity details between viewpoints. Therefore, how to compress redundant/similar coarse regions to the greatest extent without losing the fine disparity information is the greatest challenge in LF compression. Traditional compression methods [7–9] combine subaperture images of the LF image into a sequence of frames and compress them by using a video encoder. However, their compression efficiency is limited, and it is easy to lose disparity details because the high spatial and angular similarities that exist in LF images are not fully taken advantage of [10]. To take advantage of these similarities, researchers have proposed discarding a large part of the image by reducing the angular resolution during compression and reconstructing it during decompression with an LF reconstruction method, for example, angular superresolution [10]. Using this approach, a higher compression efficiency can be achieved; however, it suffers from a severe problem that is ignored by many works: the extremely poor generalization performance of reconstruction algorithms. Experiments have shown that most reconstruction algorithms have a large difference in performance on different datasets, and the reconstruction results of some datasets are seriously distorted. Therefore, the first challenge faced in this research is how to ensure the stability of the LF reconstruction algorithm when designing a new LF reconstruction-based compression scheme. It is worth mentioning that it is not expected that the stability of the LF reconstruction algorithm will be gained at the expense of excessive LF compression performance.

1.2.2 Challenge of LF Delivery

The needs of LF video transmission cannot be fully met simply by designing a new LF encoder due to two limitations. First, the complex cross-reference relationship between viewpoints leads to a lack of flexibility in the decoding process. For example, when the user is viewing the upper left area of the viewpoint matrix, it is still necessary to completely transmit all viewpoint data, even though only a part of it is truly needed because of the complex reference relationship between LF viewpoints. Second, the bitrate of LF video is much higher than that of traditional high-resolution videos, leading to LF video users observing higher distortion in the case of network impairments. Therefore, a fixed delivery scheme is not suitable for LF video.

Similar to conventional video, LF video can be transmitted either reliably, for example, with dynamic adaptive streaming over HTTP (DASH) or quick UDP internet connections

(QUIC), or with best-effort delivery such as terrestrial, satellite, cable, and internet protocol television (IPTV) video transmission systems. While over-the-top (OTT) services such as DASH, QUIC, and other reliable delivery methods have grown significantly in recent years, they constitute only 5% of global video delivery; i.e., 95% of video delivery today is achieved using terrestrial, satellite, cable, and IPTV systems [11], which are best-effort delivery methods. Therefore, the best-effort scenario is addressed in this study as interactive streaming applications, which typically use video over UDP. The second challenge to be addressed in this thesis is how to design an adaptive LF delivery scheme to provide best-effort LF delivery through real-time adjustments under different network situations while preserving the compression performance of the newly designed encoder.

1.2.3 Challenge of LF Depth Estimation

Many dedicated schemes have been proposed for LF depth estimation and have achieved good performance. However, they often suffer from the problem of insufficient generalization in practical applications. Most of these algorithms are designed for narrow-baseline LF cameras (e.g., Lytro) and are very sensitive to baseline changes in the data source. These algorithms are not compatible with wide-baseline LF acquisition systems [12, 13], which have increasingly appeared in recent years. Although a few works [14–17] have specifically proposed depth estimation algorithms for wide-baseline or sparsely downsampled LF data, they are not adaptable to narrow-baseline LF data. The existing LF depth estimation methods are only suitable for processing LF data that meet the requirements of a certain baseline range. Thus, the third challenge is to address the problem that the baseline of the LF acquisition device affects the performance of the depth estimation algorithm.

1.2.4 Challenge of the Edge-Assisted LF Framework

Similar to other internet applications (IoT (Internet of Things), automatic robotics, etc.), the advantages that MEC technology can bring to LF are as follows: First, it reduces the amount of data that the backbone network needs to transmit with the help of real-time processing at edge nodes. Second, it can offload the user's computing tasks to the edge nodes, thereby lowering the heavy burden on the user's equipment. Finally, compared with the cloud computing framework, MEC can minimize the service response delay loss caused by task offloading. Thus, the last main challenge in this study is how to abstract and mathematically model the tradeoff between the abovementioned computational and communication factors in designing the LF MEC framework and searching for the optimal strategy for task scheduling and system resource allocation.

1.3 Research Approach

The approach to address the above technical challenges and research problems is as follows:

For the challenge of LF compression, most LF reconstruction algorithms are designed for reconstruction without the reference image. However, the high-resolution reference image is indeed available during compression. If this reference image can be used to improve the reconstruction algorithm that is later used in decompression, the quality of the decompressed result can be improved. Based on this observation, graph neural network (GNN) technology could be a very good choice for reconstruction-based LF compression. A GNN model can be trained to contain similar relationships between different viewpoints of an LF video. The data of some viewpoints are then purposely discarded, and the remaining viewpoints are sent together to the GNN model. When decoding, the remaining viewpoints are used as input to run the forward GNN model, thereby recovering the discarded viewpoint data. Fewer viewpoints need to be transmitted, resulting in higher compression efficiency.

However, as previously mentioned, LF reconstruction has severe generalization issues. Experiments have shown that the main reason for this performance instability is that the reconstruction algorithm fails when dealing with the high-frequency components of the LF image. To solve this problem, a modified compression architecture, which separates the high-frequency components of the LF image from the low-frequency components and adopts different compression strategies for each, is proposed. At the same time, an estimation model is also designed to find the optimal parameter settings for the proposed scheme in different application scenarios.

For the challenge of LF delivery, best-effort video delivery is adversely affected by network packet loss. To address this type of loss, the system can use various methods such as Automatic Repeat reQuest (ARQ), forward error correction (FEC), the built-in error resilient coding (ERC) features of codecs such as H.264, and multiple description coding (MDC), in which each description is individually decodable and mutually refinable. MDC has been shown [18] to be more effective than the others for real-time and/or live video applications over networks with a moderate or high packet loss rate, such as moderately congested networks or wireless networks, the latter of which is more prone to packet loss due to interference [19]; in fact, a wireless network need not be congested to experience moderate bursts of packet loss, so MDC can be a good solution for robust video transmission over unreliable wireless networks [20]. In addition, compared with other transport layer modified solutions, in MDC, optimization only occurs at the service layer without changing the network architecture, which is very conducive to its deployment in different network environments.

In this study, a dynamic adaptive LF video transmission scheme that achieves high compression and yet provides near-distortion-free LF video when the network condition is stable is proposed. Additionally, for unstable network conditions, a description scheduling algorithm, which can decode the LF video with the highest possible quality even if partial data cannot be completely received and/or received in a timely manner, is proposed.

For the challenge of LF depth estimation, an optical-flow-based depth estimation method for uniform baseline light fields is proposed to achieve reliable depth estimation for LF data with different disparity ranges. It first divides the central SAI into patch blocks of the same pixel size and finds the optimal offset vector based on the optical flow prediction results for patches in different depth regions. According to the offset vector and the relative positional relationship between viewpoints in the angular domain, the target patch can locate the matching patch with the smallest disparity in all noncentral SAIs so that both the wide- and narrow-baseline LF data can be processed into a uniform baseline form with the same disparity range. Finally, a multiblock CNN is designed to make one-by-one predictions for all uniform patches of the central SAI, finally constructing the comprehensive depth estimation result.

For the challenge of the edge-assisted LF framework, an edge-assisted framework for LF delivery is proposed. This framework allows the LF viewpoint generation task to be offloaded between the user's equipment and the edge node, effectively reducing service response latency and the computational energy consumption of the user's equipment. At the same time, the joint offloading configuration and resource allocation problems are formalized to optimize the tradeoff between energy consumption and caching consumption under a long-term latency constraint. Finally, a scheduling algorithm to search for the optimal task offloading and allocation strategy in a multiuser scenario is developed and can achieve a good balance between computing, communicating and caching.

1.4 Research Contributions

The intended contributions of this thesis proposal are as follows:

- A novel two-layer referenced GNN-based LF compression scheme, which can achieve higher compression efficiency compared to existing schemes because it uses fewer subaperture images to recover higher quality LF images, is proposed. It can ensure the robustness of the decompression quality through a two-layer encoding architecture that treats high-frequency and low-frequency image components differently, leading to better quality compared to existing schemes.

- A multidescription separation method that can downsample the viewpoint matrix to separate the LF data with multilevel descriptions is proposed. The GNN-based LF compression method is used as the basic encoder, which enables each description to be independently decoded. The adjustable design of the basic cell makes the structure of the delivery scheme more flexible. A best-effort description transmission strategy, which can increase the possibility that users can synthesize a higher quality description, is designed.

- An offset vector estimation method based on optical flow, which can process the patches of different baselines into a unified baseline range through matching and translation in the spatial domain, is proposed. A multiblock deep convolutional neural network, which can estimate the accurate depth map of central SAI with patch offset vector, optical flow and patch disparity as input, is designed.

- An edge-assisted framework to jointly optimize the offloading configuration and resource allocation for LF multi-UE requests in an edge environment is proposed, and it explicitly considers the tradeoff between caching, service latency, and energy consumption.

1.5 Publications

The following is a list of publications generated during the course of the Ph.D. program.

Journals:

[J1]. **Hu, Xinjue**, Chenchen Wang, Lin Zhang, and Shervin Shirmohammadi. "Optical-Flow-based AI-Assisted Depth Estimation for Narrow- and Wide-Baseline LF." Submitted to IEEE Transactions on Image Processing.

[J2]. **Hu, Xinjue**, Chenchen Wang, Lin Zhang, Guo Chen, and Shervin Shirmohammadi. "Edge-Assisted Virtual Viewpoint Generation for Immersive Light Field." IEEE Multimedia. (Under minor revision)

[J3]. **Hu, Xinjue**, Yuxuan Pan, Yumei Wang, Lin Zhang, and Shervin Shirmohammadi. "Multiple Description Coding for Best-Effort Delivery of Light Field Video using GNN-based Compression." IEEE Transactions on Multimedia (2021). Early Access.

[J4]. **Hu, Xinjue**, and Lin Zhang. "Angular-spatial analysis of factors affecting the performance of light field reconstruction." IET Image Processing 16.4 (2022): 1027-1035.

[J5]. **Hu, Xinjue**, Jingming Shan, Yu Liu, Lin Zhang, and Shervin Shirmohammadi. "An adaptive two-layer light field compression scheme using GNN-based reconstruction." ACM Transactions on Multimedia Computing, Communications, and Applications 16.2s (2020): 1-23.

[J6]. Zhang, Xiaoyi, **Xinjue Hu**, Zhong Lin, Shervin Shirmohammadi, and Lin Zhang. "Cooperative tile-based 360-degree panoramic streaming in heterogeneous network using scalable video coding." IEEE Transactions on Circuits and Systems for Video Technology 30.1 (2020): 217-231.

[J7]. **Hu, Xinjue**, and Lin Zhang. "Mobile edge assisted live streaming system for omnidirectional video." Mobile Information Systems 2019 (2019).

Conferences:

[C1]. **Hu, Xinjue**, Chenchen Wang, Yuxuan Pan, Yunming Liu, Yumei Wang, Yu Liu, Lin Zhang, and Shervin Shirmohammadi. "4DLFVD: A 4D Light Field Video Dataset." Proceedings of the 12th ACM Multimedia Systems Conference. 2021.

[C2]. **Hu, Xinjue**, Jingming Shan, Yu Liu, and Lin Zhang. "Adaptive two-layer light field compression scheme based on sparse reconstruction." Proceedings of the 10th ACM Multimedia Systems Conference. 2019.

[C3]. Jinming, Shan, **Xinjue Hu**, Yu Liu, and Lin Zhang. "Adaptive Parameters Estimation for Light Field Reconstruction using Shearlet Transform." 2018 International Conference on Network Infrastructure and Digital Content. IEEE, 2018.

[C4]. **Hu, Xinjue**, Xiaoyi Zhang, and Lin Zhang. "A frame-based multi-view video transmission system for multi-user environment." 2017 17th International Symposium on Communications and Information Technologies. IEEE, 2017.

Presentation:

The results achieved in this thesis have grabbed the attention of the ISO/ITU-T JPEG (Joint Photographic Experts Group) Pleno standard group, who make the JPEG Pleno standard for representing new imaging modalities, such as texture-plus-depth, light field, point cloud, and holographic imaging. At their invitation, the following talk has been given:

- **Hu, Xinjue**, Shervin Shirmohammadi, and Lin Zhang, "AI-Assisted Light Field Depth Estimation and Compression", 2nd JPEG Pleno Workshop on Learning-Based Light Field Coding Proceedings, ISO/IEC JTC1/SC29/WG1, Online, September 21, 2022.

We hope that our work will contribute to the JPEG Pleno standard.

1.6 Thesis Organization

The rest of this thesis is organized as follows:

- In Chapter 2, background information on LF streaming and processing is presented. An extensive review of existing works in related areas will also be presented in this chapter. Material from the author's published papers [J1, J2, J3, J5, C2] are partly reused with permission in this chapter.
- In Chapter 3, the detailed design of the adaptive two-layer LF compression scheme using GNN-based reconstruction is presented. The research in this chapter corresponds to the author's published paper [J5] and material from this paper is reused with permission.
- In Chapter 4, the detailed design of the MDC best-effort delivery of LF video using the GNN-based encoder proposed in Chapter 3 is represented. The research in this chapter corresponds to the author's published paper [J3], and material from this paper is reused with permission.
- In Chapter 5, the detailed design of the optical-flow-assisted LF depth estimation method, which works well on both wide- and narrow-baseline LF data, is presented. The work in this chapter corresponds to the authors published paper [J1], and material from this paper is reused with permission.
- In Chapter 6, the detailed design of the edge-assisted virtual viewpoint generation framework for LF is presented. The research in this chapter corresponds to the authors published paper [J2], and material from this paper is reused with permission.
- In Chapter 7, a summary of this study and proposed future research directions are presented.

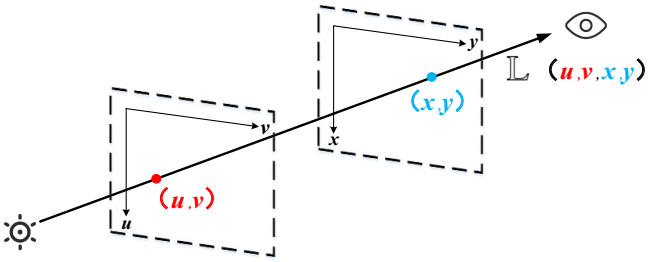
Chapter 2

Background and Related Works

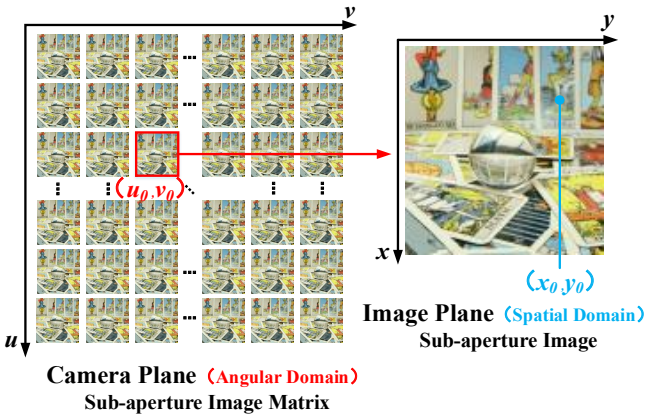
2.1 LF Basics

A light field can be represented as a plenoptic function, which was first proposed by Adelson and Bergen, who described the LF as the 7-dimensional function $\mathrm{L}(x, y, z, \theta, \phi, \lambda, t)$ [21]. Essentially, light rays are measured by the camera at every location (x, y, z) in space, at every possible angle (θ, ϕ), at every visible wavelength λ, and at every time t. However, 7-dimensional functions are hard to work with; therefore, M. Levoy et al. proposed a two-plane method to represent the LF as a 4-dimensional function [22]. As shown in Figure 2.1(a), the two-dimensional planes (u, v) and (x, y), representing the camera plane and the image plane, respectively, are parallel in space. A pair of intersection points on these two planes are created when light travels from the source to the viewer. The coordinates of these intersection points can be uniquely used to determine the direction of the light ray. The LF is further reduced to a static LF image to remove the time variable t. Additionally, wavelength λ can be represented by different color channels (such as RGB or YUV). Thus, the plenoptic function can be approximated as a 4D function that is formulated as $\mathbb{L}(u, v, x, y)$.

To record the 4D LF, early acquisition systems implemented a camera array. As shown in Figure 2.1(b), in the camera array approach, the scene is recorded with a set of lenses using different viewpoints. The image captured by each separate lens is referred to as a subaperture image, and together, all the subaperture images form an image matrix according to their corresponding lens positions. For example, $(\mathrm{u}_0, \mathrm{v}_0)$ determines the position of the subaperture image shown by the red box, and $(\mathrm{x}_0, \mathrm{y}_0)$ indicates the position of the pixel in the image, as shown by the blue dot in the figure. The camera array is the most direct



(a) Two-dimensional planes



(b) Camera array

Figure 2.1: Two plane representation and subaperture image matrix for $\mathbb{L}(x, y, u, v)$. (Reprinted from [J5], ©2020 ACM, reused with permission.)

scheme that can be used to acquire the two-plane LF; however, the cameras in the array can be too large. Furthermore, all lenses must always be kept at the same focal length. As such, a camera array is not the best solution for personal or commercial LF cameras. As a more practical solution, Ren Ng et al. proposed a handheld LF camera using the lenslet method [23]. The acquisition result of the lenslet LF camera is shown in Figure 2.2. This camera uses macro pixels, as shown by the red box, to record light information from different directions of the same light source. In this context, subaperture refers to a single pixel in a macro pixel. By decoupling macro pixels into regular pixels and recombining them, the subaperture image matrix can be obtained, as shown by the blue box. Lenslet and camera arrays use the same principle, but the former can be made smaller and does not need to separately consider the focal length of the lenses. Therefore, current handheld commercial LF cameras and, subsequently, current LF datasets, mostly adopt the lenslet scheme. Hence, the transmission and processing of the image matrix is the focus of this

study, as explained below.

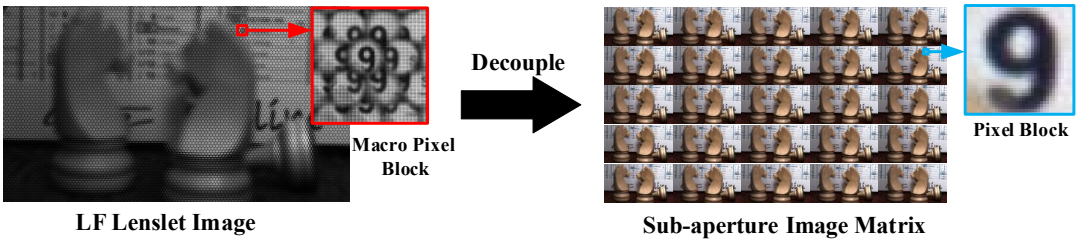


Figure 2.2: Lenslet LF camera. (Reprinted from [J5], ©2020 ACM, reused with permission.)

In the LF matrix, by switching between existing viewpoints or synthesizing new viewpoints, users can move freely in both the vertical and horizontal directions. At the same time, the dense viewpoints of LF are very beneficial for the extraction of depth information of the scene. Therefore, the positional movement of users in the forward-backward direction can also be supported. As shown in Figure 2.1(b), the lens plane (u, v) is regarded as the angular domain of the LF signal because each subaperture viewpoint in (u, v) can be seen as the result obtained from different viewing angles. Accordingly, the angular resolution of LF is the number of subaperture images. The denser the angular domain is, the more scene details are captured and recorded. The image plane (x, y) is regarded as the spatial domain of the LF signal. The spatial domain resolution of the LF is the pixel resolution of each subaperture image.

Usually, the coordinate (u, v) corresponds to the location of the viewpoint in the microlens array, and the coordinate (x, y) corresponds to a pixel in the subaperture image of a single viewpoint. $\widehat{\mathbb{L}}(w_x, w_y, w_u, w_v)$ characterizes the 4D Fourier spectrum of the LF. $\mathbb{L}_{u,v}(x, y)$ denotes the 2D spatial slice of the LF, which is actually the 2D subaperture image of each viewpoint. $\widehat{\mathbb{L}}_{u,v}(w_x, w_y)$ corresponds to the 2D Fourier spectrum of the spatial slice $\mathbb{L}_{u,v}(x, y)$. Similarly, $\mathbb{L}_{x,y}(u, v)$ represents the angular slice of the LF.

2.2 LF Compression and Streaming

Much effort has already been dedicated to the study of LF compression and streaming. The common LF compression schemes can be roughly divided into three categories:

In the first category of schemes, the conventional image and video coding schemes, such as JPEG, JPEG 2000, H.264 and H.265, are modified to compress the LF image. In an

early work, Aggoun et al. [24] designed an improved scheme based on 3D-DWT for JPEG 2000 to effectively utilize the correlation between SAIs and further improve the compression ratio in the spatial dimension. Liu et al. [25] proposed a coding scheme that predicts subaperture views layer by layer. The views in the lower layers were used as the reference to predict views in the higher layers. L. Li et al. [7] proposed a 2D hierarchical coding scheme to reduce the occupancy of the reference buffer. They considered LF images as a pseudosequence consisting of multiple frames with slightly different viewpoints and divided subaperture images into four quadrants. The quadrants were compressed using standard video encoders, such as H.264/AVC and H.265/HEVC. Conti et al. [8] proposed a new spatial prediction model based on self-similarity compensated prediction. The new prediction model can improve the standard H.265/HEVC scheme to satisfy the specific characteristics of 3D holoscopic content. Lucas et al. [26] proposed a new way to improve the performance of 3D video encoding by using locally linear embedding techniques. Monteiro et al. [27] introduced bipredicted self-similarity estimation and self-similarity compensation into H.265/HEVC to compress LF images. Y. Li et al. [9] proposed a bidirectional spatial prediction model to modify the interprediction process using H.265/HEVC. This new method can adapt to the characteristics of the LF image well. Perra et al. [28] proposed a lossy compression scheme for unfocused LF images. The raw LF data were compressed by JPEG 2000 after demosaicing and slicing. Finally, Zhang et al. [29] presented a new LF representation based on point clouds, named Surface LF, which maps traditional LF images onto a 3D point cloud and represents each subaperture image by a B-spline wavelet. Then, any point cloud compression scheme can be applied to compress the LF. While approaches in the first category of schemes are easy to develop and implement, they do not take advantage of the large redundancies between subaperture images.

In the second category of schemes, redundancy is utilized to design new LF-specific compression schemes. Perra et al. [30] proposed a coding scheme based on low-complexity preprocessing of the raw LF image; a pseudotemporal sequence of frames was produced in advance for standard compression using HEVC. Zhang et al. [31] designed an adaptive prediction structure for LF compression using the scale-invariant feature transform (SIFT) algorithm to determine the prediction structure. Guo et al. [32] built a mathematical model to illustrate the combination of weighted distortion and visual consistency and then used a convex optimization algorithm to find the optimal LF compression parameters. Amirpour et al. [33] proposed a method to first decompose the LF image into multiple subaperture images, and then, these images were scanned in a snake order. This method can efficiently exploit the redundancy between adjacent subaperture images to increase the compression ratio. Hariharan et al. [34] introduced circular reordering and performed low-complexity preprocessing before compression. Dansereau et al. [35] proposed a way to encode a subset

of the LF image after devignetting, demosaicing and alignment. Kundu et al. [36] proposed a new LF compression scheme using homography and 2D warping to predict views. Finally, Jiang et al. [37] used low-rank approximation to modify LF compression. For this category of schemes, there is a noticeable improvement in compression compared to that of the first category.

The third category of schemes uses the LF reconstruction algorithm for LF compression. Originally, LF reconstruction was designed to perform angular superresolution for sparsely sampled LFs to restore the full light field using limited samples. However, LF reconstruction can also be applied to LF compression by deliberately discarding parts of the LF data, hence achieving significant compression and recovering the lost data through reconstruction. Xu et al. [38] presented a reconstruction algorithm that can restore 4D LF from a portable LF camera without the need for calibration images by demosaicing based on kernel regression to enhance the quality of the reconstructed subaperture images. Shi et al. [1] considered LF reconstruction as a sparsity optimization problem in the continuous Fourier domain and recovered the full LF from a small number of subaperture images on the 1D viewpoint trajectories formed by 4 edges and 2 diagonals. The functional flow of Shi's reconstruction algorithm is shown in Figure 2.3. Initially, to simulate the undersampled LF image, the complete LF image was resampled according to the 1D trajectory. After resampling, the algorithm performed a 2D Fourier transform on each spatial slice (w_x, w_y) (i.e., the subaperture image) of the LF image and converted these slices into an angular slice set (w_u, w_v) in the order of angular domain coordinates. Next, each angle slice was densely reconstructed in two steps using the SFFT method. First, the position of the significant frequency component of the angular slice was roughly located by projection and voting. Then, the approximate position was taken as the initial point, and the gradient search algorithm accurately found the coordinates of each frequency component. Hawary et al. [39] described an LF scalable compression scheme based on the research by [1]. They downsampled raw LF images according to the 1D trajectory during encoding. When decoding, they used reconstruction algorithms to recover complete LF data.

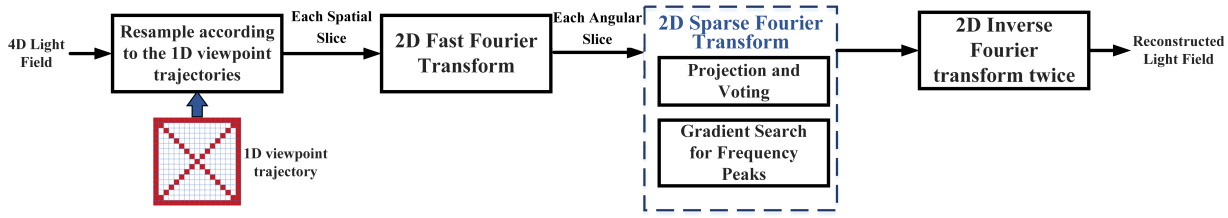


Figure 2.3: LF sparse reconstruction algorithm proposed in [1]. (Reprinted from [J5], ©2020 ACM, reused with permission.)

Continuing in the third category, Viola et al. [40] proposed representing the correlation between LF images and a graph model. They constructed a graph, in which each vertex represents one view, and learned the edges' modeling relationships between corresponding views. Then, a subset of these views was selected to be compressed, whereas the remaining views were recovered using the graph. To obtain a graph to best describe the relationships among the views while emphasizing signal smoothness on its structure, Viola et al. established the following optimization problem:

$$\begin{aligned} \underset{W \in \mathcal{W}_m}{\arg\min} \quad & tr(Y^T L Y) - \alpha \mathbf{1}^T log(W\mathbf{1}) + \beta \|W\|_F^2 \\ & L = D - W \\ & \mathcal{W}_m = \{W \in \mathbb{R}_+^{m \times m} : W = W^T, diag(W) = 0\} \end{aligned} \tag{2.1}$$

where W is a weight matrix that uniquely describes a graph (L is the graph Laplacian matrix). Y is an LF image vectorized in such a way that each row represents one entire view. m is the total number of views. The parameter α controls connectivity in the graph, and β controls the sparsity of the graph. The gradient search algorithm is used to solve the above problem. The accuracy of the graph model depends largely on the settings of parameters α and β. However, these two parameters are greatly dependent on the experience in the actual application, and the best settings greatly differ for different LF images. Therefore, in this study, the graph neural network [41] is considered as another good way to quickly and automatically build the graph model for the image matrix. In Chapter 3, a new reference GNN-based reconstruction algorithm with a dedicated two-layer encoding architecture for LF compression is proposed. This algorithm makes the proposed solution significantly better in terms of compression performance, as will be shown later.

2.3 Multiple Description Coding (MDC) Basics

In this section, an overview of the basic concepts of MDC is presented.

MDC is a robust coding technique that is used for video transmission through error-prone networks. In MDC, video is encoded into multiple descriptions with a certain degree of redundancy between the descriptions. Each description is self-sufficient, i.e., it can be decoded and displayed. Typically, each description represents a subset of the video frame that has less information than the original frame. All descriptions together will give the complete original video frame. Figure 2.4 shows the design of MDC with two descriptions. There are three decoders that can be selected during decoding. The central decoder is used

when all descriptions are correctly received. Otherwise, the corresponding side decoder is used to decode the correctly received description.

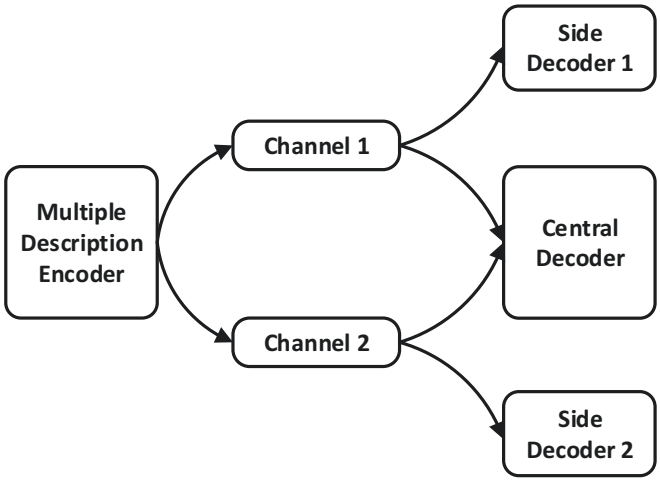


Figure 2.4: Multiple description coding with two descriptions. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

If an MDC scheme is correctly designed, even though the receipt of one description is sufficient for the receiver to be able to display an image to the viewer with sufficient quality, receiving more descriptions will add more quality. MDC can enhance the stability, reliability, and visual quality of conventional video streaming systems. For example, Kazemi et al. [42] studied the problem of finding the optimum amount of MDC redundancy, which has a critical role in MDC performance. They proposed a scheme in which the redundancy budget is allocated to the frames based on the weighted mismatch-rate slopes so that the additional bitrate can attain maximum distortion reduction, leading to higher video quality. In [43], Kazemi et al. presented their investigation results for a new intracoding approach in MDC. They developed a cost function based on which intra/interframe type is selected to find the most suitable I-frame positions within a given video stream, leading to optimum compression.

Recently, MDC has been used for 3D/multiview streaming. For example, Lin et al. [44] proposed a region-based multiple description coding scheme for robust multiview-plus-depth-based 3D video communication, in which two descriptions are formed by setting the left and right views as dominant in the first and second descriptions, respectively. Cheng et al. [45] proposed an MDC-based method to compress and transmit a 3D point cloud to a receiver over wireless or internet communication channels. In their method,

every superpoint in the point cloud is encoded with two descriptions: the codes of the median feature vectors that reconstruct a simplified 3D model and the codes to encode the residual feature vectors of superpoints. Finally, in [46], Chen et al. proposed an efficient multiple description coding scheme for multiview video based on the correlation of spatial polyphase transformed subsequences. This scheme can be used to build a robust 3D video transmission system for error-prone channels.

Inspired by the good results of the above work in utilizing MDC for 3D/multiview video delivery, in Chapter 4 of this study, MDC is used to design a dynamic and adaptive LF video delivery scheme. This scheme uses the viewpoint as the unit to downsample the LF video and subdivide it into different levels of description. Then, the optimal sending sequence is determined by the scheduling algorithm. The proposed scheme can provide higher video quality at the receiver under the premise that all LF video view streams can be decoded and restored.

2.4 LF Acquisition System

With the development and evolution of LF acquisition equipment in recent years, the impact of baseline changes in LF data sources on the design of subsequent processing or application algorithms has become increasingly difficult to ignore. As shown in Figure 2.5, the baseline represents the physical spatial distance between two adjacent lenses. Since the lens arrays of LF acquisition equipment are distributed at equal intervals, the interval B of the lens array is the standard baseline of the acquisition equipment. The baseline B plays a very important role in LF depth estimation. For example, the light source O in Figure 2.5 is imaged into different pixels P_1 and P_2 through two vertical adjacent lenses. Due to the angle difference between the two lenses and the light source O, the spatial coordinates of P_1 and P_2 in their respective SAIs are not consistent, and this coordinate difference is called disparity. Since the aspect ratio of most lenses is not equal to 1, the disparity is usually not the same in the horizontal and vertical directions, which are defined as D_s and D_t, respectively. As shown in the lower part of the figure, the vertical light source imaging process is used as an example. The vertical disparity D_t, the depth distance of the light source D, the baseline B and the focal length F of the lens conform to a proportional relationship, which can be expressed by Equation (2.2).

$$\frac{D_t}{D} = \frac{B}{F} \tag{2.2}$$

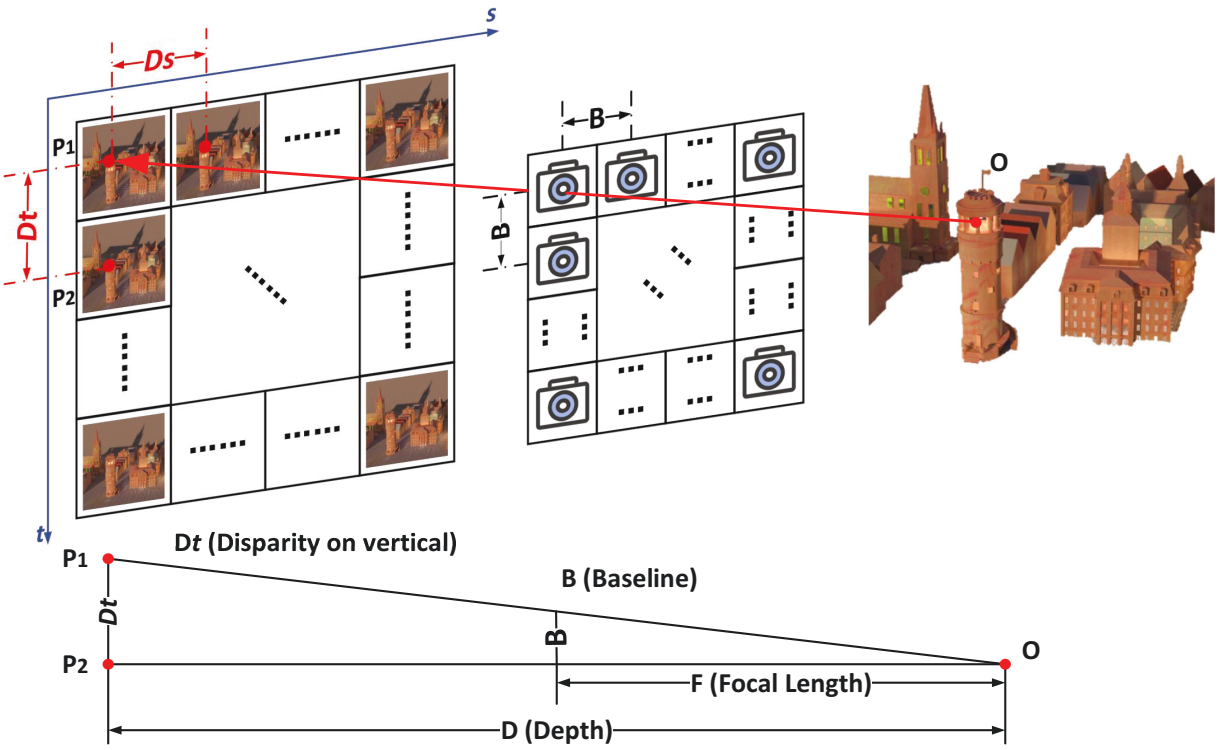
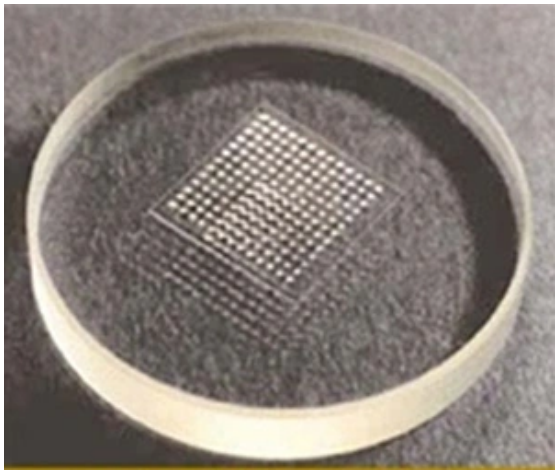


Figure 2.5: LF camera imaging model. (Reprinted from [J1], submitted to IEEE for peer review.)

Therefore, depth estimation based on multiview images (not only the LF) requires the detection of disparity. When the baseline B of the LF acquisition device becomes wider, the disparity of the acquired SAI matrix also proportionally increases. Therefore, LF acquisition devices are divided into two categories according to their baseline distribution ranges: narrow-baseline and wide-baseline ranges.

Lytro LF cameras are the main representatives of narrow-baseline LF cameras. All Lytro products are designed based on lenslet technology [23], so their baselines are approximately 4 mm, which is considered narrow. The lenslet can integrate a microlens array on a small chip, and all microlenses share the same photosensitive element, as shown in Figure 2.6(a). Figure 2.6(b) shows the Lytro R8 Raytrix, an LF camera that supports video capture. The number of shooting viewpoints is 5×5, and the parallax spacing between microlenses (baseline) is only 4 mm.

Although a narrow-baseline LF camera improves portability, it suffers from significant disadvantages in terms of the field of view, focal length, and production cost, which makes



(a) Lenslet matrix on a glass substrate

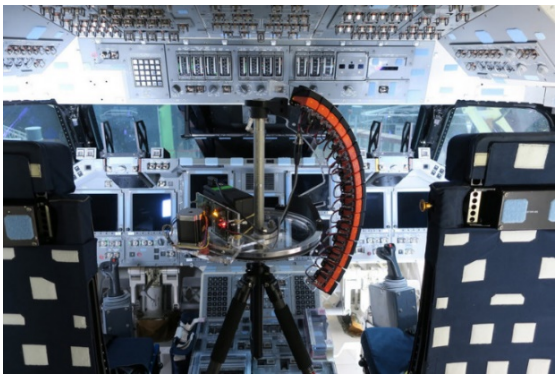
(b) Lytro R8 Raytrix

Figure 2.6: Narrow-baseline LF camera.

it disadvantageous when compared with RGBD and stereo cameras. The development trend of LF acquisition technology changed after Google acquired Lytro, and camera array systems suitable for one-time large-scale acquisition are now increasingly favored. In 2018, Google modified the GoPro Odyssey Jump camera platform, bent it into 16 lenses distributed in vertical arcs and mounted it on a rotating platform, as shown in Figure 2.7(a). This rotating LF acquisition system realizes a spherical LF video array acquisition system through time-lapse shooting. It takes approximately 1 minute for the camera platform to rotate and record approximately 1000 outward viewing angles on a 70 cm sphere.

In 2020, M. Broxton et al. of the Google LF team proposed a spherical acquisition array and a supporting processing algorithm for LF video [12], as shown in Figure 2.7(b). They captured LF video using a custom array of 46 time-synchronized cameras distributed over the surface of a dome in a hemisphere with a diameter of 92 centimeters, while a backend viewpoint interpolation algorithm was used to improve close areas of the scene. This solution realizes all-around perception of the shooting scene through the spherical shooting array and simultaneously captures video from each viewpoint to realize the perceptual modeling of dynamic scenes. Compared with the narrow-baseline LF camera based on lenslet technology, the baseline of the above LF acquisition systems using the camera array scheme is larger by an order of magnitude and considered to be a wide baseline.

Correspondingly, the disparity change in SAIs caused by the widening of the baseline is very significant. The detailed zoomed-in patch of the LF image *Toy* is taken as an example. There are two different versions of the LF image *Toy*, a narrow-baseline version



(a) Google's rotary acquisition platform in 2018

(b) Google's spherical acquisition array in 2020

Figure 2.7: Wide-baseline LF acquisition system.

and a wide-baseline version. Both versions were acquired using a 5×5 lens array, and the physical space position and orientation of their center lens were exactly the same, but different baseline ranges were used for acquisition. Figure 2.8 shows the central SAI of the LF image *Toy*, and the red box represents the part of the patch (pixel size is 32/*times*32) that it will be zoomed-in on for comparison in Figure 2.9. Figure 2.9(a) shows the zoomed-in details of the local patches from the narrow-baseline version of *Toy*. Its pixel disparity distribution ranges from -1.65 pixels to 1.61 pixels, where the signs represent the disparity direction. Taking the horizontal axis as an example, the negative sign means that the disparity is to the left. Figure 2.9(a) shows patches from the wide-baseline version. Due to the wider baseline, its disparity range is enlarged to between -13.55 pixels and 12.61 pixels. As seen from Figure 2.9, the overlap ratio between different lenses of the narrow-baseline LF is very high, so their SAI image content is highly homogenized. This is very beneficial for LF depth estimation, as homogeneity helps to establish correlations between pixel blocks across different SAIs. However, in wide-baseline LF, the relative positions between pixel blocks corresponding to different depth planes and different objects change drastically, which makes it very difficult to locate them among all SAIs. This is why many depth estimation algorithms that work well on narrow-baseline LF fail on wide-baseline LF. This thesis only focuses on LF data collected by parallel lens or camera matrix, and both narrow- and wide-baseline LF data are involved.

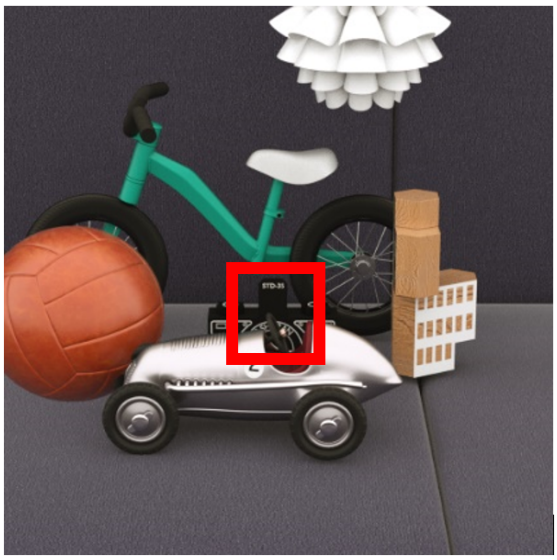


Figure 2.8: Central SAI of the LF image *Toy*. The zoomed-in patch of the red box is $32px \times 32px$ and is used in Figure 2.9. (Reprinted from [J1], submitted to IEEE for peer review.)

2.5 LF Depth Estimation

As a 6-DOF media, LF has great potential for usage in 3D environments, immersive or otherwise. Depth estimation is one of the core components in a 3D modeling pipeline. Compared with the classic multiview (binocular or trinocular) depth estimation scheme, LF depth estimation has the following three advantages.

First, the prediction results are more accurate mainly because multiview-based depth estimation relies on feature point matching. In many cases, due to the pixel color difference between viewpoints or the occlusion relationship between objects, it is difficult to find the correct matching feature points between different viewpoints using binocular schemes; therefore, the accurate disparity corresponding to the feature point cannot be estimated very well. However, due to the sufficient number of viewpoints in LF and the distribution of viewpoints at equal intervals, the imaging pixel positions of the same light source in different SAIs will present a regular linear distribution in the 4D space, as shown in Figure 2.10. As shown in the figure, when the LF SAI matrix $L(u, v, s, t)$ is transformed into the su plane of the epipolar plane image (EPI), the pixels corresponding to the same light source point form a continuous straight line in the EPI image [47]. The slope of the line reflects the depth of the light source from the acquisition camera, and the greater the

(a) Narrow-baseline version $(-1.65px, 1.61px)$

(b) Wide-baseline version $(-13.55px, 12.61px)$

Figure 2.9: Zoomed-in patches from different baselines of the LF image *Toy*. (Reprinted from [J1], submitted to IEEE for peer review.)

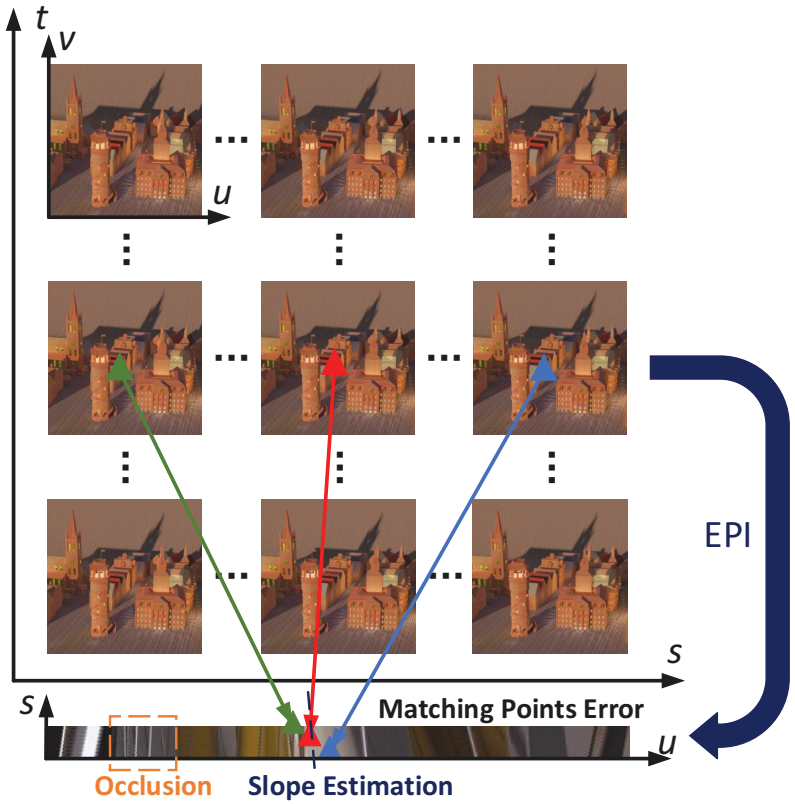


Figure 2.10: Epipolar plane image (EPI) and its matching feature point. The example EPI image is transformed by the SAI matrix: $s = 5$, $u = 310$. (Reprinted from [J1], submitted to IEEE for peer review.)

slope is, the greater the depth, and vice versa. Compared with matching only the feature points in three viewpoints, represented by triangles of different colors in Figure 2.10, the slope estimation method for the EPI image can more easily eliminate the interference of various errors and noises.

Second, LF depth estimation can be more robust to the influence of occlusion of objects in the scene. As shown in the orange dotted box of the EPI image in Figure 2.10, the slashes with different slopes overlap each other. The upper and lower sections of some of them are not complete, and a certain part in the middle is crossed and covered by other slashes. As a result, LF depth estimation is not completely invalid when the feature points of a few viewpoints are lost. Even if some slopes in the EPI image are not continuous, a relatively accurate slope estimation can still be obtained.

Third, compared to binocular or monocular images, the LF has great potential in recognizing and reconstructing non-Lambertian planes. Non-Lambertian plane mainly refers to specular reflection, such as water surface and glass. The main reason for the difficulty of depth estimation and recognition of the non-Lambertian plane is that the nonuniform reflection characteristics of the light source on the non-Lambertian plane cause the chromaticity of the imaging pixels to vary discontinuously between different SAIs. However, this does not break the coordinate continuity of the space-angle domain after EPI transformation, which makes it possible to further analyze the optical reflection properties of non-Lambertian plane surfaces after depth estimation [48].

Much research work on LF depth estimation has been published, but as previously discussed, most of this research is designed for narrow-baseline LF cameras, and the issues introduced by wider baselines of LF acquisition equipment are not addressed. For example, Jeon et al. [49] proposed a cost-function-based multiview stereo matching method to achieve depth estimation with subpixel accuracy to solve the problem of depth estimation for narrow-baseline light-field images. Zhang et al. [50] designed a rotating parallelogram operator to measure the slope of the EPI line, improving the accuracy of EPI-based depth estimation in occluded and noisy scenes. Sheng et al. [51] designed a comprehensive cost function that combines the EPI in the horizontal, vertical, and diagonal directions and obtained a depth estimation result with high accuracy. Shin et al. [52] introduced a fast and accurate LF depth estimation method based on a fully convolutional neural network. Li et al. [53] proposed an end-to-end fully convolutional network to estimate depth values at intersections on horizontal and vertical EPI images.

With an increase in the number of wide-baseline LF acquisition systems, some researchers have noticed the new challenges brought by the large-disparity LF, i.e., sparse LF, for LF depth estimation. Leistner et al. [14] introduced the concept of EPI shift and designed a neural network architecture to solve the problem of depth estimation for wide-baseline LF. Jiang et al. [15] proposed an algorithm for disparity estimation, which only needs to use a sparse set of dense LF as input, for each viewpoint considering occlusions. Shi et al. [16] proposed a coarse to fine depth estimation method, which uses a learning-based approach to adaptively adjust the search accuracy by region. This algorithm can also provide basic coarse-grained depth estimation results for sparse-LF inputs. Chuchvara et al. [17] proposed a depth estimation algorithm based on superpixels for sparse, wide-baseline LF data. In their source, the input SAIs are oversegmented into nonoverlapping compact superpixels. They then modeled superpixels as planar patches and used them as the basic unit for depth estimation. This algorithm maximizes the preservation of the contour integrity of the scene objects in the depth results.

Most of the above depth estimation algorithms designed for wide-baseline and sparse

LF have two limitations: the first limitation is that the algorithm requires prior information on the range of disparity, and the algorithm needs to input a suitable EPI shift value in advance to work. The second is that the pixel-block-based method can scale the range of disparity but at the expense of granularity. This causes them to not work well with narrow-baseline LF data. In Chapter 5 of this study, a uniform baseline processing method based on optical flow, which can adaptively process LF data of different baselines into a uniform disparity distribution range according to the optical flow prediction results, is proposed. The design based on patch segmentation enables the proposed method to consider the influence of objects in different depth regions on the disparity range, further improving the accuracy and generalization of the algorithm.

2.6 Multi-accses edge computing and LF viewpoint interpolation

Much work on viewpoint interpolation for LF has been proposed [54]. Vagharshakyan et al. [55] proposed a shearlet filter to upsample the LF in the EPI domain, which can leverage the rich structural information of EPIs to synthesize the new LF viewpoint. Panner et al. [56] proposed an LF view synthesis method for the representation of multiplane images (MPIs). They encoded depth as a soft visibility map into the multilayer alpha channel to assist in synthesizing the new viewpoint of any position with any focal length. Yue et al. [54] presented a benchmark collection for LF view interpolation methods. Any of the above methods can be utilized as a virtual viewpoint generation method in the system proposed in Chapter 6.

However, the limited computing power of user LF devices may not meet the requirements of various high-complexity viewpoint interpolation algorithms. So the last part of this thesis also wants to introduce edge computing architecture to reduce the burden of user devices while maintaining low service delay. There are many edge computing frameworks focused on video streaming and processing. Wang et al. [57] theoretically modeled a scenario where multiple video users request different CNN models on edge servers for real-time processing. Sun et al. [58] proposed an edge-computing-based framework for mobile 360 degrees video delivery. They optimized the offload options for different users to meet the field of view (FOV) projection needs. However, unlike the assumptions in previous works, the required computational resource for the LF task is usually not the same on UE and EN because of the differences in viewpoint caches on different platforms, which result in requiring different interpolation rounds for virtual viewpoint rendering. This more complex linkage between the viewpoint cache and rendering computation inspired us to design

a more efficient model for an edge-based viewpoint rendering framework for immersive LF, as shown in Chapter 6.

Chapter 3

An Adaptive Two-Layer Light Field Compression Scheme Based on GNN-based Reconstruction

In this chapter, an architectural overview of the proposed scheme is first presented. Then, the proposed GNN-based reference LF reconstruction method is explained in complete detail. Finally, the two-layer encoding architecture is introduced.

3.1 Architectural Overview of the Proposed Scheme

Figure 3.1 shows the architecture of the proposed scheme, which consists of two main stages, compression and decompression. They are performed on two components: low-frequency and high-frequency components. In the compression stage, the *parameter estimation model* module will dynamically adjust the controllable parameters according to the application requirements. The low- and high-frequency components are then separated. The former are used to train the GNN graph model and then downsampled. After independently encoding the high-frequency components and downsampling the low-frequency components with HEVC, both types of components will be packaged together with the graph model and transmitted. In the decompression stage, after unpackaging and independently HEVC-decoding the low- and high-frequency components, the LF reconstruction method uses the graph model to reconstruct the low-frequency components. Combined with the high-frequency components, a high-quality image can then be recovered.

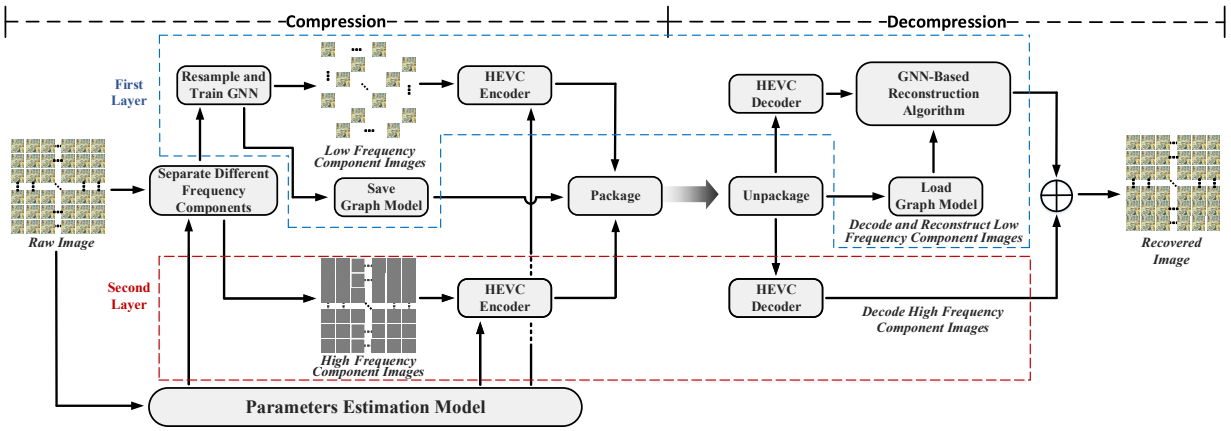


Figure 3.1: Adaptive two-layer LF compression scheme based on GNN-based reconstruction. (Reprinted from [J5], ©2020 ACM, reused with permission.)

Next, the GNN-based reference reconstruction algorithm will be presented, and as previously mentioned, the reasons for its recovery distortion will be discussed.

3.2 GNN-based Reference LF Reconstruction

3.2.1 Design of the Reconstruction Algorithm

Because of the special data structure of the image matrix, using vertices and edges to represent different viewpoints and their correlations in the image matrix, as shown in Figure 3.2, is a promising idea. Combined with the GNN, a more efficient and adaptive technique to determine an accurate graph model for the image matrix can be obtained. As shown in Figure 3.2, the LF graph model can be represented as $G_{LF}(V, E)$, where V is a set of vertices v. Each vertex represents a subaperture image normalized as a one-dimensional vector, and E is a set of edges e, where each edge between two vertices represents a reference relationship between the two viewpoints. $\mathcal{N}(v)$ represents the neighborhood of node v, which is composed of all adjacent nodes around v.

Each node v is characterized by its feature x_v with a ground truth label R_v. $x_{(v,u)}$ is used to represent the feature of the edge between node v and node u. For the input anchor viewpoint, their input features are the same as their labels, which are one-dimensional vectors flattened by the original SAIs. For the viewpoint waiting for reconstruction, its input feature is copied from the nearest anchor viewpoint SAI directly. The hidden state h_v

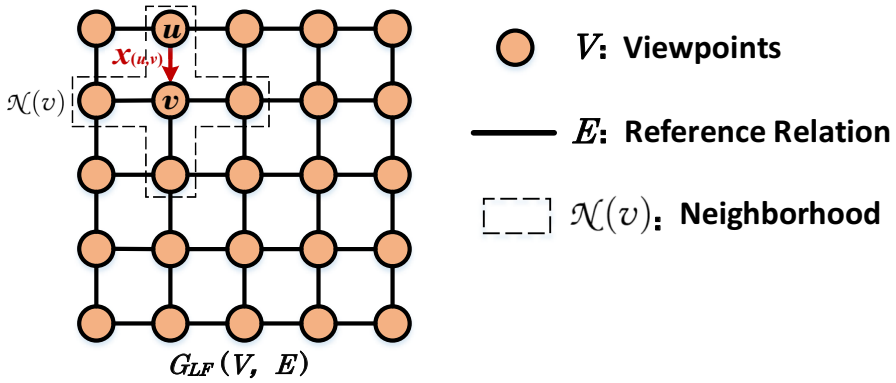


Figure 3.2: Graph for the image matrix. (Reprinted from [J5], ©2020 ACM, reused with permission.)

(state embedding) for node v contains information from other nodes in $\mathcal{N}(v)$. The hidden states of all nodes make up the graph model, and the goal of the GNN is to determine the approximation of all h_v. The times t and $t+1$ represent the current and next iteration steps in the GNN training process, respectively. Finally, to ensure that each node can perceive other nodes on the graph, the GNN iteratively updates the hidden state of all nodes [59] until the network model converges. At time $t+1$, the hidden state of node v is updated as follows:

$$h_v^{t+1} = f(x_v, x_n[\mathcal{N}(v)], x_{(v,n)}[\mathcal{N}(v)], h_n^t[\mathcal{N}(v)]) \tag{3.1}$$

where f is the update function for the hidden state, which is also known as the local transaction function. $x_n[\mathcal{N}(v)]$ refers to the features of all neighbor nodes. $x_{(v,n)}[\mathcal{N}(v)]$ refers to the features of all edges adjacent to node v. $h_n^t[\mathcal{N}(v)]$ refers to the hidden state of the neighbor node at time t. Function f works on all nodes in the graph and is a globally shared function.

During training, the function f is repeatedly used to update the hidden state of each node until the entire graph model becomes stable. Then, each node can perceive the information of all other nodes in the graph. In this process, a loss function is also needed to continuously modify the function f so that the final stable state of the graph can be closer to the ground truth. It can be defined as:

$$loss = \frac{1}{|V|} \sum_{v \in V} (-R_v \cdot log(h_v^t)) \tag{3.2}$$

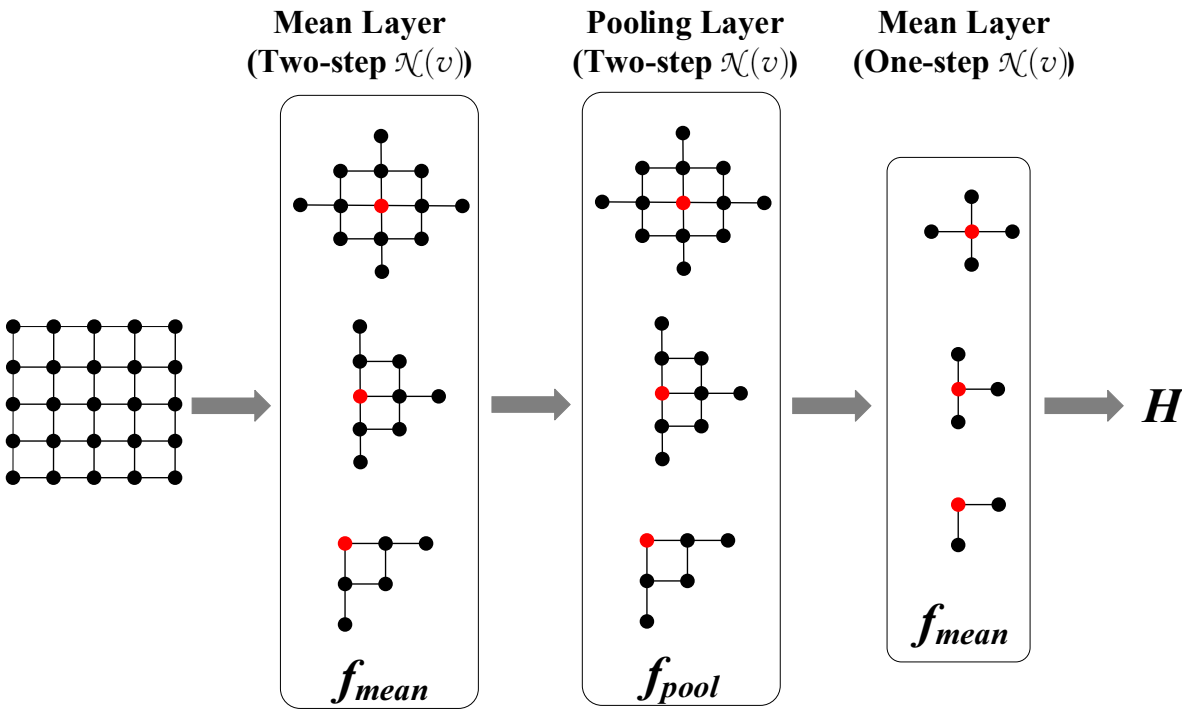


Figure 3.3: GNN model for LF reconstruction. (Reprinted from [J5], ©2020 ACM, reused with permission.)

Similar to traditional deep neural networks, GNNs can be cascaded through multiple different types of layers. As shown in Figure 3.3, the network model used in this chapter consists of two mean layers and one pooling layer. The output H in the figure consists of the hidden state h of all nodes.

As defined in Equation (3.3), the pooling aggregator proposed in [60] is used as the f in the pooling layer.

$$f_{pool} = max(\{ReLU(W H_n^t + b), \forall n \in \mathcal{N}(v)\}) \tag{3.3}$$

where max denotes the elementwise max operator, and $\mathcal{N}(v)$ is defined according to the step distance between nodes. For example, the two-step $\mathcal{N}(v)$ represents a set of all nodes that can be connected to node v through two adjacent edges. The larger the range of $\mathcal{N}$ is, the stronger the node's ability to perceive the global graph, but the slower the convergence speed. For one layer, $\mathcal{N}$ does not change at each iteration.

In the mean layer, the mean aggregator is defined as Equation (3.4):

$$f_{mean} = W \cdot Mean(\{h_v^t\} \cup \{h_n^t, \forall n \in \mathcal{N}(v)\}) \tag{3.4}$$

The selection of $\mathcal{N}$ is the same as in Equation (3.3).

Usually, the mean aggregator is rough and linear. Although it cannot directly and accurately fit complex nonlinear local transaction functions very well, it can help nodes quickly perceive the information of surrounding nodes, which is very helpful for the accelerated convergence of the overall network. In the pooling layer, by applying the max pooling operator to each computed feature, the model can be nonlinearly fitted. The proposed network does not use as many layers as in traditional deep neural networks, mainly because deeper GNN models cannot be achieved through simple cascading. Since the number of nodes in the graph and the vector dimensions of the hidden state are the same between different layers, cascading between layers is essentially equivalent to performing more iterations. The purpose of setting the third layer in the proposed model is to reduce the neighborhood size of some mean iteration steps to control the consumption of computer memory during network training.

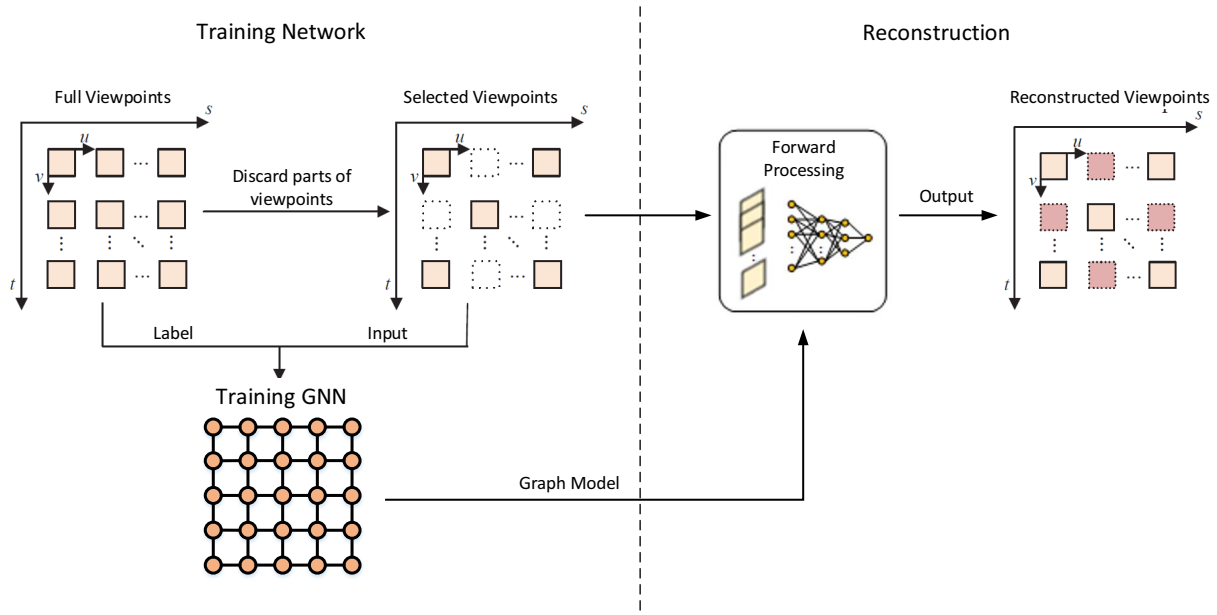


Figure 3.4: Flowchart for GNN-based reference LF reconstruction. (Reprinted from [J5], ©2020 ACM, reused with permission.)

The overall flowchart of the GNN-based reference LF reconstruction algorithm is shown

in Figure 3.4. When extracting the graph model from the full viewpoints of LF, only a subset of the viewpoints will be retained and selected. The grid in Figure 3.5 represents the angular plane of the LF image, the red square represents the viewpoint that is selected and retained, and the white square indicates that the position data are discarded. When the raw LF image matrix has $n \times n$ subaperture images and the angular domain sampling interval is k, the proportion of the retained viewpoints after resampling is:

$$q_{lp} = \frac{\lfloor n^2/k \rfloor + 1}{n^2} \tag{3.5}$$

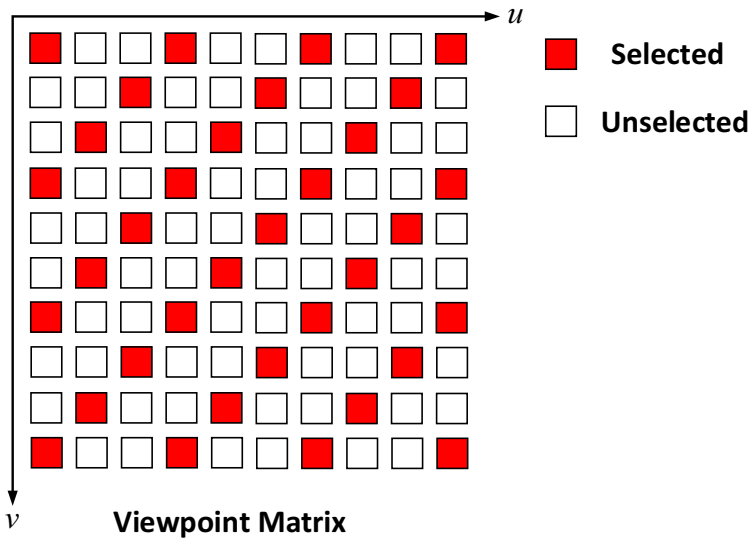


Figure 3.5: Matrix of the discarded viewpoints during compression. (Reprinted from [J5], ©2020 ACM, reused with permission.)

In fact, the sampling interval k can be freely selected, and the sparse sampling matrix will result in poorer reconstruction results. In this chapter, k is set to 3 to achieve a balance between the data compression ratio and reconstruction quality.

The selected viewpoints are used as the input data, and the full viewpoints are used as the labeled data to train the GNN. In the graph model, before iteration, the initial state of a retained viewpoint can be directly obtained, while the initial state of a discarded viewpoint is copied from the nearest node. After training, the generated network containing the graph model will be transmitted and used during decompression; i.e., the reconstruction algorithm uses the graph model and the received viewpoints as input and runs the forward

processing network to reconstruct all viewpoints, including the discarded viewpoints.

3.2.2 Performance of the Reconstruction Algorithm

The proposed reconstruction algorithm and some other schemes are tested on different LF images. As shown in Figure 3.6, five classic datasets (*Jelly Beans, Bunny, Flower, Stanford Crystal* and *Knight*) were tested using three reconstruction algorithms: the proposed GNN algorithm, the sparse fast Fourier transform (SFFT) scheme in [1], and the graph-based learning (GBL) approach in [40]. Each row represents the reconstruction result of a different dataset using the same reconstruction method. Compared with the SFFT reconstruction used in the author's previous work (MMsys 2019) [C2], the quality of the results of the GNN is improved by an average of 5 dB. This is mainly because the application scenarios of LF compression and LF reconstruction are not exactly the same. During the reconstruction process, the actual high-resolution LF images are unknown, so they cannot be used as a reference in the SFFT reconstruction algorithm. However, in the compression process, this problem does not exist. The proposed scheme and the GBL-based scheme can fully mine the information from the original high-resolution LF image to improve decompression. In addition, the performance of the GNN scheme is better and more stable than that of the GBL scheme. This is because the local transaction function f of the hidden states obtained by the GNN can better represent the relationship between viewpoints in the LF graph compared to the simple edge weight matrix W used by the GBL. Furthermore, since the gradient search algorithm is sometimes polarized, the performance of GBL is unstable.

Another advantage of the proposed scheme is that it transmits fewer viewpoints than the other schemes to reconstruct the complete image matrix. As shown in Figure 2.3, the SFFT sampling matrix is composed of bounding boxes and diagonals. For an LF image with 13×13 viewpoints, $12 \times 4 + 11 \times 2 - 1 = 69$ viewpoints need to be retained. In the GBL scheme, only when a missing viewpoint has valid viewpoints around it can the missing viewpoint be recovered based on the graph's edge weight matrix. Therefore, the GBL scheme must retain at least half the number of viewpoints to work, i.e., $13 \times 13/2 = 85$ viewpoints. According to Figure 3.5, the number of input viewpoints required for the proposed scheme is only $\lfloor 13^2/3 \rfloor + 1 = 57$.

Compared with the other two schemes, the proposed GNN scheme can transmit fewer viewpoints and achieve better reconstruction quality, leading to better coding efficiency. Since the reconstruction results of the three schemes on the *Jelly Beans* and *Bunny* datasets are all very good, these two images are used for comparison among the 3 LF compression schemes. As shown in Figure 3.7, the rate-distortion curve of the proposed scheme is obviously better than the other two and therefore more suitable for LF compression.

Angular Resolution 13×13 Spatial Resolution 1024×1024	 Jelly Beans	 Bunny	 Flower	 Stanford Crystal	 Knight
SFFT	PSNR:43.2763 SSIM:0.9966 VMAF:95.4270	PSNR:40.1273 SSIM:0.9492 VMAF:89.5998	PSNR:34.5894 SSIM:0.8983 VMAF:84.5897	PSNR:27.8248 SSIM:0.8389 VMAF:82.2541	PSNR:26.6423 SSIM:0.8272 VMAF:58.5599
GBL	PSNR:45.1774 SSIM:0.9934 VMAF:95.1729	PSNR:41.5678 SSIM:0.9680 VMAF:92.5516	PSNR:35.1774 SSIM:0.9034 VMAF:83.1729	PSNR:23.6421 SSIM:0.7715 VMAF:66.6673	PSNR:29.6421 SSIM:0.8415 VMAF:72.8417
Proposed GNN	PSNR:44.6411 SSIM:0.9920 VMAF:97.0729	PSNR:44.2475 SSIM:0.9987 VMAF:95.6800	PSNR:38.1165 SSIM:0.9445 VMAF:87.6522	PSNR34.7149 SSIM:0.8739 VMAF:87.0080	PSNR:32.2792 SSIM:0.8432 VMAF:86.1330

Figure 3.6: Reconstruction results on datasets using three algorithms. (Reprinted from [J5], ©2020 ACM, reused with permission.)

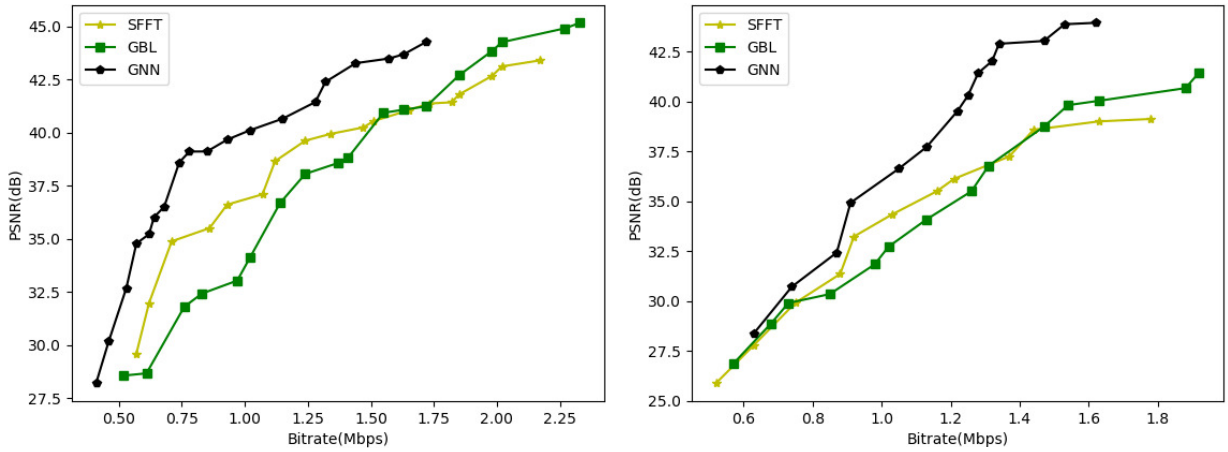


Figure 3.7: Rate-distortion curves for three schemes on two datasets: (a) *Jelly Beans*; (b) *Bunny*. (Reprinted from [J5], ©2020 ACM, reused with permission.)

However, Figure 3.6 also shows that there is a very large difference in performance among different datasets. From the experimental results, the results on *Jelly Beans* and *Bunny* are better, and the peak signal-to-noise ratio (PSNR) of the reconstructed image exceeds 40 dB. However, the reconstruction algorithm performed worse on *Knight* and *Stanford Crystal*, with PSNRs lower than 30 dB, while the performance on *Flower* was intermediate. We can further see that the structural similarity index (SSIM) of the results is also basically consistent with the PSNR. To measure the subjective performance of the reconstruction algorithm, the results were also evaluated by Netflix's VMAF. According to [61], VMAF, which has been trained and tested with a large amount of user data on Netflix severs, can reflect the subjective feeling that the image quality brings to the user. We can see that the VMAF scores range from 0 to 100, with 100 indicating the best subjective experience. Since the three reconstruction schemes have the greatest difference in the quality of the results on *Knight* and *Stanford Crystal*, their visual examples are shown in Figure 3.8.

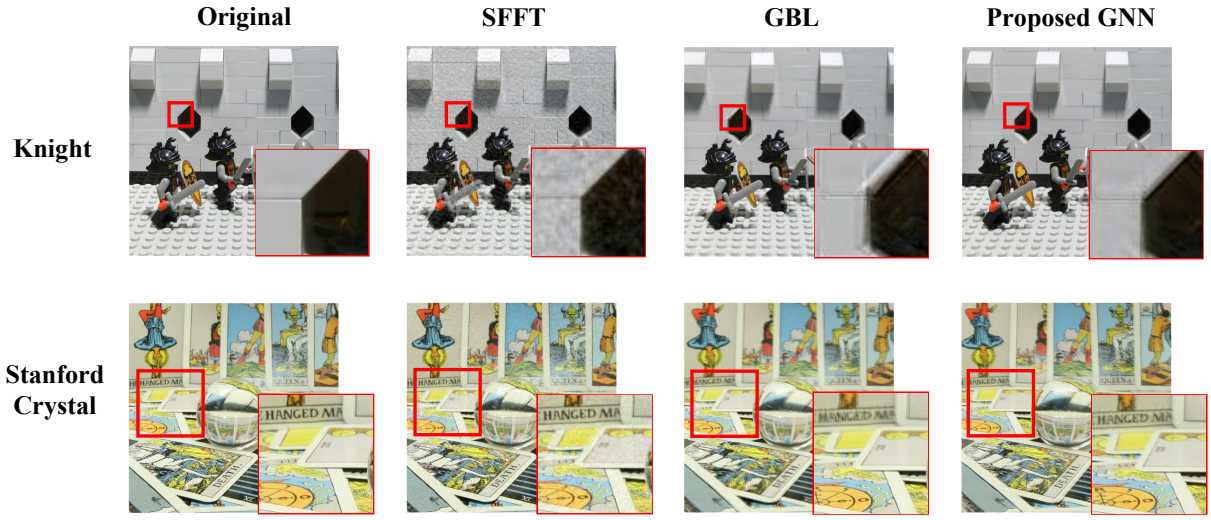


Figure 3.8: Visual comparison of the results using three reconstruction schemes on *Knight* and *Stanford Crystal*. (Reprinted from [J5], ©2020 ACM, reused with permission.)

Because the performance of the reconstruction algorithm on different datasets greatly differs, this algorithm cannot be directly applied to the compression scheme. The visual examples also confirm the instability of the performance of the reconstruction algorithms on different datasets. Therefore, the cause of this performance instability must be found first. Based on further analysis of the results, it can be found that the contents of images that lead to better reconstruction results are relatively simple, such as those in the *Jelly*

Beans and *Bunny* datasets. In contrast, images with more complex scene information, such as those in *Stanford Crystal* and *Knight*, have poorer reconstruction performance. Therefore, it can be reasonably assumed that the degree of distortion in the reconstruction process is related to the content of the image. In other words, it is mainly affected by the representation of the LF signal in the spatial domain. Fast Fourier transform is performed on the subaperture images for the five datasets in both horizontal and vertical directions, and the results are shown in Figure 3.9. We can see that the images in *Knight* and *Stanford Crystal* contain more high-frequency components than those in *Jelly Beans* and *Bunny*. Hence, it can be hypothesized that performance instability is due to the differences between low-frequency and high-frequency components in the datasets.

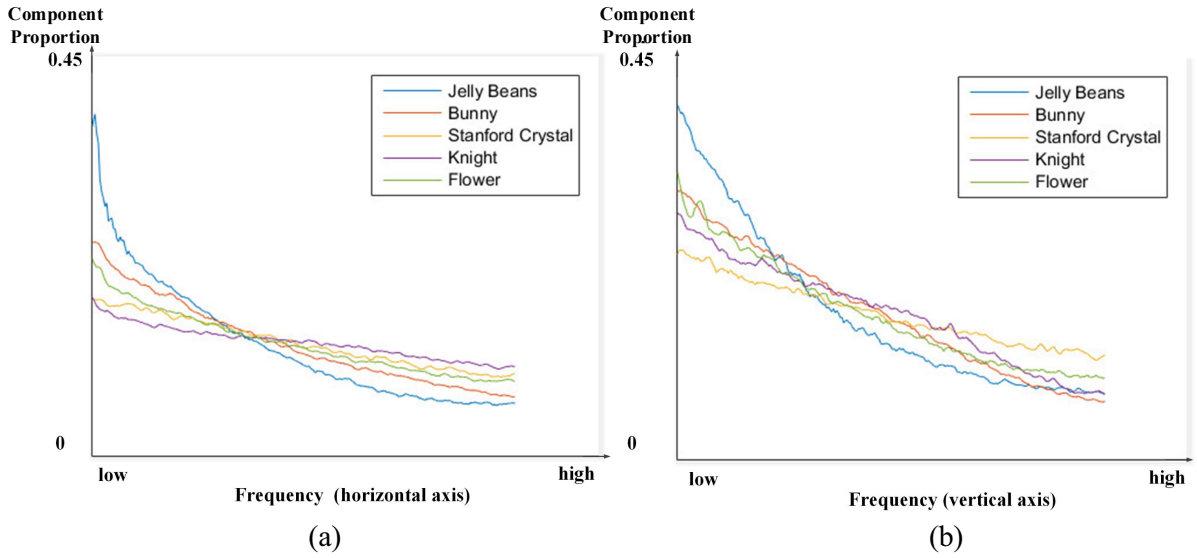


Figure 3.9: Fourier analysis on different datasets: (a) horizontal direction of the subaperture image, and (b) vertical direction of the subaperture image. (Reprinted from [J5], ©2020 ACM, reused with permission.)

To verify this hypothesis, the LF image is tested in blocks to more directly reflect the result. Each viewpoint of each dataset is divided into 128×128 blocks. Because the spatial resolution of the dataset is 512×512, each block's resolution will be 4×4. All blocks are separately processed by the reconstruction algorithm. The matrix P contains the PSNR results for all blocks and is normalized to (0,1). The visualization results of the P matrices for *Stanford Crystal*, *Knight* and *Bunny* are shown in Figure 3.10. The darker the pixel is, the worse the reconstruction quality of the block. The test results show that the reconstruction quality of the edge and texture regions is significantly lower than that

of the other regions. Therefore, it can be assumed that these high-frequency components are the main cause of reconstruction instability.

Figure 3.10: Visual results of the P matrices for *Stanford Crystal*, *Knight* and *Bunny*. (Reprinted from [J5], ©2020 ACM, reused with permission.)

To further verify this assumption, a specific verification experiment was designed. The *Stanford Crystal*, on which the algorithm performed poorly, was chosen as the subject. The raw LF image and the reconstructed result were both passed through a low-pass filter with the same bandwidth ratio of the passband, and the PSNR was used to judge the distortion of the reconstruction result in the current frequency band.

Specifically, for the fixed angular plane coordinates (x, y), separate spatial slices $\mathbb{L}_{u,v}(x, y)_{raw}$ and $\mathbb{L}_{u,v}(x, y)_{rec}$ are first obtained from the raw image matrix and the reconstructed image matrix, respectively. Then, fast Fourier transform (FFT) is simultaneously performed on both matrices. The diagonal length of the spectral image after FFT is regarded as the bandwidth B of the full frequency band, and then a series of low-pass filters with different passbands αB are set, where α is the bandwidth ratio of the passband and is adjusted between 0 and 1. By adjusting the value of α, it is possible to control how many high-frequency components remain after the input image passes through the low-pass filter. The PSNR is calculated to measure the degree of distortion of the result. Figure 3.11 shows the curve of the bandwidth ratio of passband α and the PSNR of the algorithm reconstruction result. We can see that as α increases, the PSNR decreases. Therefore, it can be concluded that the reconstruction algorithm's instability is related to the high-frequency components, which is also consistent with the results in Figure 3.6 and Figure 3.9. To address this, a two-layer encoding architecture is proposed, as described below.

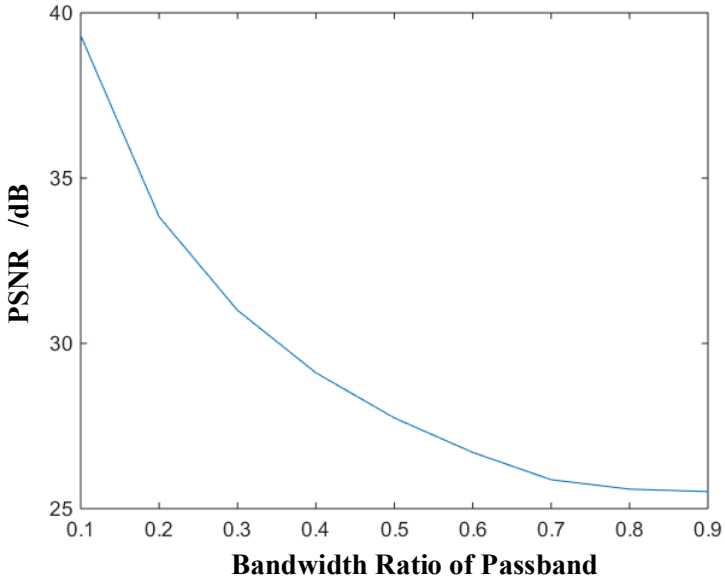


Figure 3.11: Curve of the bandwidth ratio of passband α and the PSNR of the algorithm reconstruction result. (Reprinted from [J5], ©2020 ACM, reused with permission.)

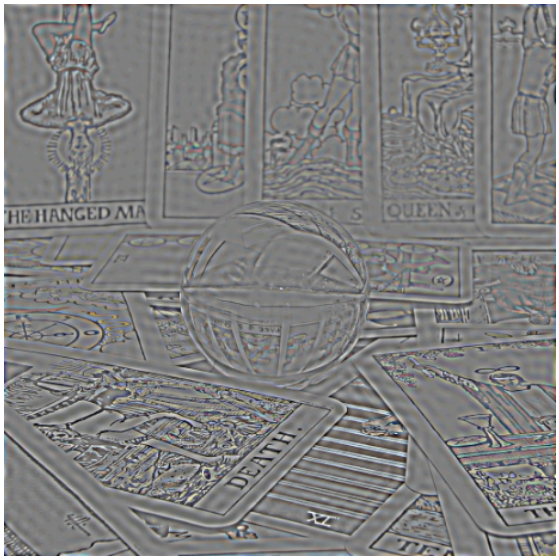
3.3 Two-Layer Encoding Architecture

3.3.1 Separation of the Image Frequency Components

It can be seen that the reconstruction algorithm introduces high-frequency noise into the results, while the low-frequency components are hardly affected. Therefore, to improve the performance, a dynamic two-layer LF compression scheme is proposed to separate the high-frequency and low-frequency components and independently compress them. In other words, a dynamic low-pass filter controlled by the bandwidth ratio of the passband α is needed, so most of the high-frequency components of the image can be filtered out. This not only effectively reduces the reconstruction noise but also controls the balance between the compression ratio and the quality of the decompressed image. The image of the low-frequency components I_{lp} is shown in Figure 3.12(a) for *Stanford Crystal*.

The removed image after low-pass filtering is I_{hp}, and it contains the high-frequency components of the LF, as shown in Figure 3.12(b). I_{hp} is defined as shown in Equation (3.6).

(a) Low-frequency components of subaperture image I_{lp}



(b) High-frequency components of subaperture image I_{hp}

Figure 3.12: High- and low-frequency components of the SAI. (Reprinted from [J5], ©2020 ACM, reused with permission.)

$$I_{hp} = \begin{cases} I_{raw} - I_{lp} & \text{where } |I_{raw} - I_{lp}| > T_h \\ 0 & \text{where } |I_{raw} - I_{lp}| < T_h \end{cases} \tag{3.6}$$

where I_{raw} is the raw subaperture image, and T_h is the threshold used to control how much data in I_{hp} will be saved. The larger T_h is, the more high-frequency components are removed. Conversely, a smaller T_h will result in an increase in the amount of data for I_{hp}. The above method allows us to dynamically separate the high-frequency and low-frequency components in the LF image and to compress them by different strategies, as described below.

3.3.2 Dynamic Two-Layer Compression Structure

In the first layer, the compression strategy for low-frequency components is considered. For I_{lp} of all subaperture images in the LF image matrix, the data of partial subaperture viewpoints are discarded during the compression process and recovered through reconstruction during decompression. As shown in Figure 3.5, whether to discard a low-frequency

component image I_{lp} is decided based on its angular plane coordinates (u, v). After resampling the I_{lp} matrix of the LF, the set of reserved I_{lp} is encoded with HEVC. Qp_{lp} is the quantization parameter selected when encoding the I_{lp} sequence.

In the second layer, the compression strategy for high-frequency components is considered. Since applying the reconstruction algorithm to high-frequency components is the main cause of decompression noise, a conservative compression strategy is adopted for matrix I_{hp}, whereby all its viewpoints are directly encoded with HEVC. The compression ratio and quality can be controlled by dynamically adjusting the quantization parameter Qp_{hp} and threshold T_h. As shown in Equation (3.6), the threshold T_h will make matrix I_{hp} become sparse, which is beneficial with respect to improving the compression efficiency when compressing matrix I_{hp}. However, it should be noted that the data lost by T_h are unrecoverable. In this work, T_h is set lower than 30 to help speed up the calculation of the parameter estimation model, as explained next.

3.3.3 Parameter Estimation Model

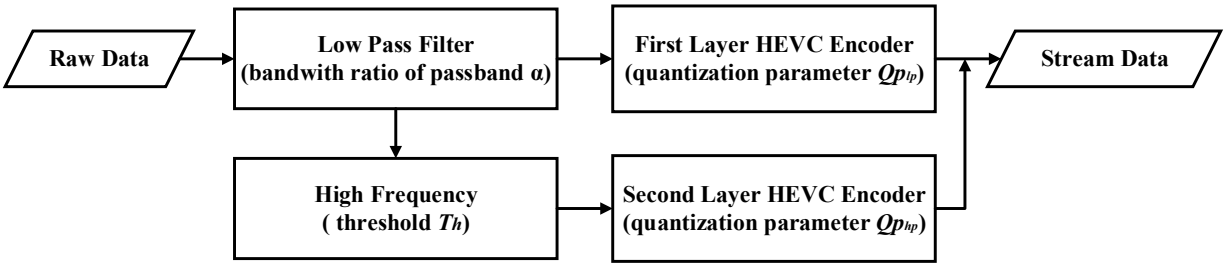


Figure 3.13: Controllable parameters in the proposed scheme. (Reprinted from [J5], ©2020 ACM, reused with permission.)

Figure 3.13 shows the 4 controllable parameters that affect compression, namely, α, T_h, Qp_{lp}, and Qp_{hp}. As shown in Figure 3.1, the *parameter estimation model* module can control the compression quality and efficiency through the adjustment of these 4 parameters. A mathematical model is designed to describe the relationship between these parameters and the compression performance to precisely tune the proposed scheme. The PSNR and the *Bitrate* of the decompression results are used as indicators to measure the performance of the schemes. The mathematical model can be derived as follows:

$$(\mathrm{PSNR}, Bitrate) = F(\alpha, \mathrm{Qp}_{lp}, T_h, \mathrm{Qp}_{hp}) \tag{3.7}$$

where PSNR is the average PSNR of all subaperture images of the decoded result, which is defined in Equation (3.8):

$$\begin{aligned} \mathrm{PSNR} &= \sum_{(m,n)\in(u,v)} 10log_{10}(\frac{255^2}{MSE(m,n)}) \\ MSE(m,n) &= \frac{1}{ij}\sum_{(i,j)\in(x,y)} \|\mathbb{L}_{m,n}(i,j)_{rec} - \mathbb{L}_{m,n}(i,j)_{raw}\| \end{aligned} \tag{3.8}$$

According to Figure 3.11, the larger the bandwidth ratio of passband α is, the more reconstruction noise is preserved, which means that the quality of the recovered I_{lp} is worse and the amount of data passing through the low-pass filter is greater; thus, we have:

$$\begin{aligned} \mathrm{PSNR}_{lp} &\propto G_1\left(1/\alpha\right) \\ D_{lp} &\propto M_1\left(\alpha\right) \end{aligned} \tag{3.9}$$

where M_1 and G_1 represent any monotonic function relationship, which may be linear or nonlinear, and D_{lp} represents the amount of data of the low-frequency components to be compressed. Because the total data amount of the raw LF is constant (defined as D_{total}), the high-frequency components to be compressed can be expressed as $D_{hp} = D_{total} - D_{lp}$. A portion of the viewpoint data after resampling according to the 1D trajectory (shown in Figure 3.5) is discarded based on D_{lp}, and then, these data are encoded using an HEVC encoder with a quantization parameter of Qp_{lp}. It can be roughly assumed that the data compression ratio of the HEVC encoder only relates to Qp, which is $R(\mathrm{Qp})$. In this model, the selection of the quantization parameters of the HEVC encoder tends to be conservative to reduce the impact of the encoding process on the compression scheme results. This can effectively reduce the complexity of the model and make the performance of the compression scheme more controllable. Therefore, after the first layer of the proposed scheme is compress, the bitrate of the low-frequency components is:

$$Bitrate_{lp} = \left(q_{lp} \times D_{lp} \times R(\mathrm{Qp}_{lp})\right)/T \tag{3.10}$$

where q_{lp} is defined in Equation (3.5), and T is the time constant for calculating the bitrate.

Similarly, a larger T_h means that more high-frequency components are discarded and the quality of recovery of I_{hp} is worse; thus, we have:

$$\mathrm{PSNR}_{hp} \propto G_2\left(1/\alpha\right) \tag{3.11}$$

In addition, as T_h increases, the proportion of data reserved in D_{hp} decreases. Therefore, this proportion can be defined as q_{hp}, and it satisfies:

$$q_{hp} \propto M_2\left(1/T_h\right) \tag{3.12}$$

M_2 and G_2 have similar functions as M_1 and G_1, respectively. Therefore, the bitrate of the high-frequency part can be defined as:

$$Bitrate_{lp} = \left(q_{hp} \times D_{hp} \times R(\mathrm{Qp}_{hp})\right)/T \tag{3.13}$$

According to the above model, the function of the parameter estimation model can be considered as solving an optimization problem, which means the optimal parameter settings $(\alpha, \mathrm{Qp}_{lp}, T_h, \mathrm{Qp}_{hp})$ can be found for each value of the target PSNR to minimize the bitrate. For example, for a target PSNR S_{PSNR} to be achieved, the following optimization problem can be obtained:

$$\begin{aligned} \min \quad & Bitrate = Bitrate_{lp} + Bitrate_{hp} \\ s.t. \quad & A(\mathrm{PSNR}_{lp}, \mathrm{PSNR}_{hp}) = S_{\mathrm{PSNR}} \\ & 0 \leq \alpha \leq 1, \\ & T_h \geq 0, \\ & 25 \leq \mathrm{Qp}_{lp}, \mathrm{Qp}_{hp} \leq 40, \end{aligned} \quad . \tag{3.14}$$

where A represents the quality of the complete recovery result, which is the weighted average of PSNR_{lp} and PSNR_{hp}. The upper limit of the selectable range of the HEVC's Qp is set to 40, and the lower limit is set to 25 to reduce the influence of the encoding on the compression result. G_1, G_2, M_1, and M_2 are all set to linear functions to simplify the model complexity; they need to be fitted by actual data. Then, the two-layer scheme with some random parameter settings is tested in advance, and the data are collected and used to fit the default function. The final optimal settings can be obtained through the optimization algorithm. In the proposed scheme, since the controllable parameters are all within a certain range, a global search algorithm is used as the optimization algorithm of the parameter estimation model. In an actual scenario, according to the application's demand for quality versus compression, the *parameter estimation model* module gives the optimal settings $(\alpha, \mathrm{Qp}_{lp}, T_h, \mathrm{Qp}_{hp})$ to control the other modules in the scheme.

3.4 Decompression Process

Similar to the compression scheme, the high-frequency and low-frequency components are decompressed separately. As shown in Figure 3.1, the low-frequency content is HEVC-decoded, and then the complete I_{lp} matrix is recovered through the reconstruction algorithm using the graph model. The high-frequency content will also be HEVC-decoded to directly obtain the I_{hp} matrix. Finally, the low-frequency and high-frequency matrices are added according to the position of the viewpoint to obtain a complete subaperture matrix of the decompressed LF.

3.5 Experimental Evaluation

3.5.1 Experimental Setup

A set of comparative experiments was designed to verify the performance of the proposed two-layer GNN-based compression scheme (`GNN-based (2-layer)`) compared to 5 others that represent state-of-the-art schemes. The first scheme, `all-view-HEVC`, encodes all subaperture images directly using HEVC. The second scheme, `SFFT-based`, uses the SFFT-based reconstruction algorithm [1] to recover the complete LF during decompression. The third scheme, (`GBL-based`), uses the graph-based learning reconstruction approach [40]. The fourth scheme, (`MMsys19_hu`), is the author's previous work [C2], and it uses the SFFT reconstruction algorithm and a two-layer coding architecture to compress the LF image. Finally, the fifth scheme, (`GNN-based (no SEPA)`), is the same reference GNN-based reconstruction algorithm but without the two-layer architecture. The schemes were run on either MATLAB 2017b or Python 3.61, while HM 11 was used as the HEVC software. The Global Optimization Toolbox in MATLAB was used in the *parameter estimation model* module to solve the optimization problem. The Deep Graph Library (DGL) and PyTorch were used to implement the reference GNN-based reconstruction algorithm. For the LF matrix with an angular resolution of 13×13 and a spatial resolution of 1024×1024, the number of trainable parameters of its GNN model is 758785, and the GPU memory required for loading its model is 920 MB. The forward prediction execution time required for single-frame decoding on the NVIDIA GeForce GTX-1080-Ti is 0.74 seconds, and the time required for model convergence in training is about 5 hours. Finally, the *Jelly Beans, Bunny, Flower, Stanford Crystal* and *Knight* datasets [62], each containing 13×13 viewpoints with a subaperture image resolution of 1024×1024, were used for testing.

3.5.2 Experimental Results

The performance of the *parameter estimation model* module is first tested. The partial parameter estimation results are shown in Table 3.1. Each row in the table contains the target PSNR and optimal parameter settings.

Table 3.1: Partial parameter estimation results.

S_{PSNR}/dB	α	T_h	Qp_{lp}	Qp_{hp}
29.0	0.50	30	30	37
29.5	0.44	26	29	37
30.0	0.42	25	30	36
30.5	0.38	33	29	36
31.0	0.34	34	31	35
31.5	0.32	23	32	35
32.0	0.30	20	32	33
32.5	0.26	14	32	33
33.0	0.24	24	32	32
33.5	0.22	24	32	32
34.0	0.20	13	31	31
34.5	0.20	12	32	31
35.0	0.18	13	33	29
35.5	0.16	13	32	29

It can be seen from the table that as the target PSNR increases, the scheme reduces the bandwidth ratio of the α passband. This shows that the *parameter estimation model* module correctly attempts to remove high-frequency components for layer 1 compression, thereby improving the quality of decompression. Additionally, the smaller the threshold T_h is, the less loss of high-frequency components, which is consistent with the T_h column in the table. The low-frequency quantization parameter Qp_{lp} is relatively stable, but the high-frequency quantization parameter Qp_{hp} decreases with increasing quality. This is because when the target PSNR is low, the main factor affecting the quality of the result is the low-frequency image, and the influence of the high-frequency image is small. Therefore, both Qp_{hp} and T_h can be set to larger values to achieve higher compression. When the target PSNR is high, the effect of the low-frequency image on the result is already very limited, so it is necessary to improve the quality of the high-frequency image as much

as possible to meet the higher PSNR requirement. It is worth noting that the proposed scheme mainly relies on reducing α to achieve the adaptation of complex LF images. This is reasonable because a smaller α means that more high-frequency details are preserved, which helps improve the quality of the decompression results.

To verify the reliability of the above results, 14 sets of parameter settings are applied in Table 3.1 in the proposed scheme to observe the actual PSNR. The estimated PSNR and actual PSNR of the 14 sets of parameter settings are compared in Figure 3.14.

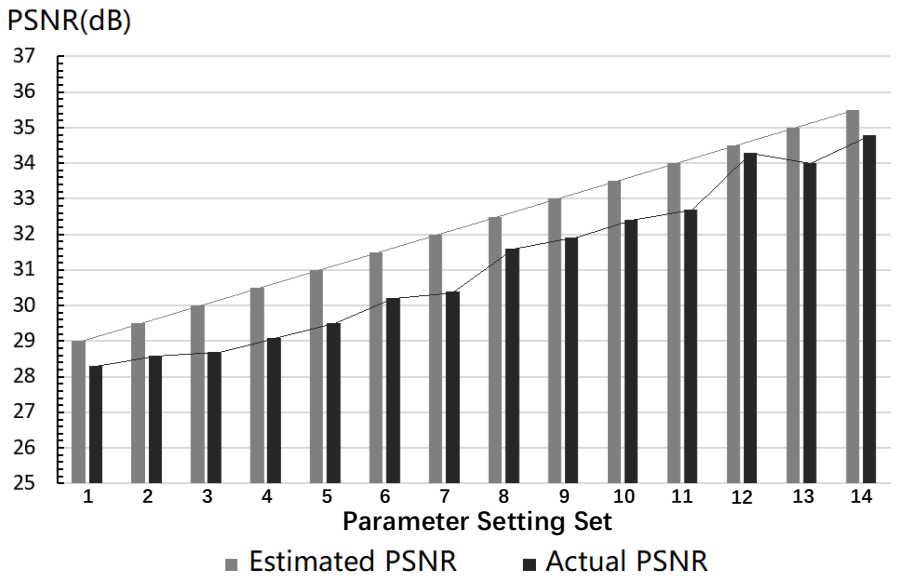


Figure 3.14: Comparison between the estimated PSNR and actual PSNR of the parameter estimation module. (Reprinted from [J5], ©2020 ACM, reused with permission.)

As shown in Figure 3.14, the actual PSNR is generally lower than the estimated PSNR, and the average error is approximately -1 dB. Although the estimated results are not completely accurate, the growth trend of the actual PSNR is in line with expectations, so the estimated parameter settings can still be used to control the proposed scheme. The main reason for this error is that it is difficult to use the general regression model to characterize the relationship between the HEVC coding quality and the quantization parameters. In particular, this relationship is related to the content of the coded image and dynamic changes. Nonetheless, since the effects of HEVC coding are limited in the proposed scheme, this degree of estimation error is within an acceptable range.

Figure 3.15 shows a comparison of the decompression results with no frequency sepa-

ration and with the proposed 2-layer architecture. The first row is the original image, a subaperture image, which serves as an example. The second and third rows are the results of the best image quality that can be achieved by the above two compression schemes. As we can see, a noticeable improvement in image quality can be achieved using the two-layer scheme, especially on the *Flower, Stanford Crystal* and *Knight* datasets, where the PSNRs of the optimal results are 7 dB better. This shows that the proposed two-layer architecture can significantly overcome reconstruction instability. Figure 3.15 also shows the SSIM and VMAF metrics. The two-layer architecture has a more stable performance.

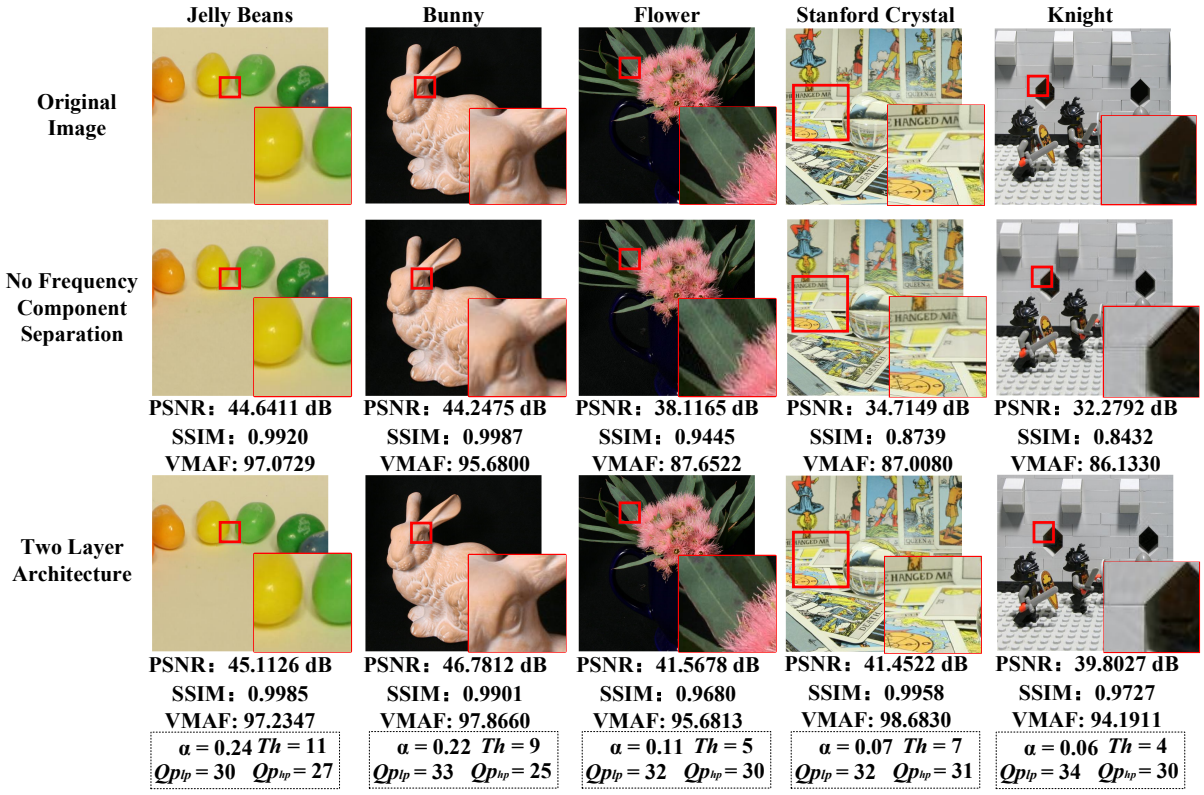


Figure 3.15: Results with no frequency separation and with the proposed 2-layer architecture. (Reprinted from [J5], ©2020 ACM, reused with permission.)

Figure 3.16 shows the rate-distortion curves of the proposed method, the GNN-based (2-layer) method, compared with the other schemes described in Section 3.5.1. The blue curve represents the proposed scheme, the `GNN-based (2-layer)` method, with the parameters shown in Table 3.2 and Table 3.3, and it has superior performance compared to other schemes, which cannot overcome the high-frequency noise of the reconstruction process, no

matter how much bitrate is used. On *Flower*, although it is less affected by reconstruction noise, the proposed scheme can still exceed the upper limit of the PSNR of the `GNN-based (no SEPA)` scheme and maintain good compression performance after the target PSNR of 38 dB. The above results prove that the proposed scheme has better generalization performance in processing LF images with different complexities. Compared with previous work [C2], which is marked in the figure as `MMsys19_hu`, the new scheme proposed in this chapter has better compression efficiency under most quality requirements. This is mainly due to the performance improvement of the referenced GNN reconstruction algorithm compared to the SFFT algorithm.

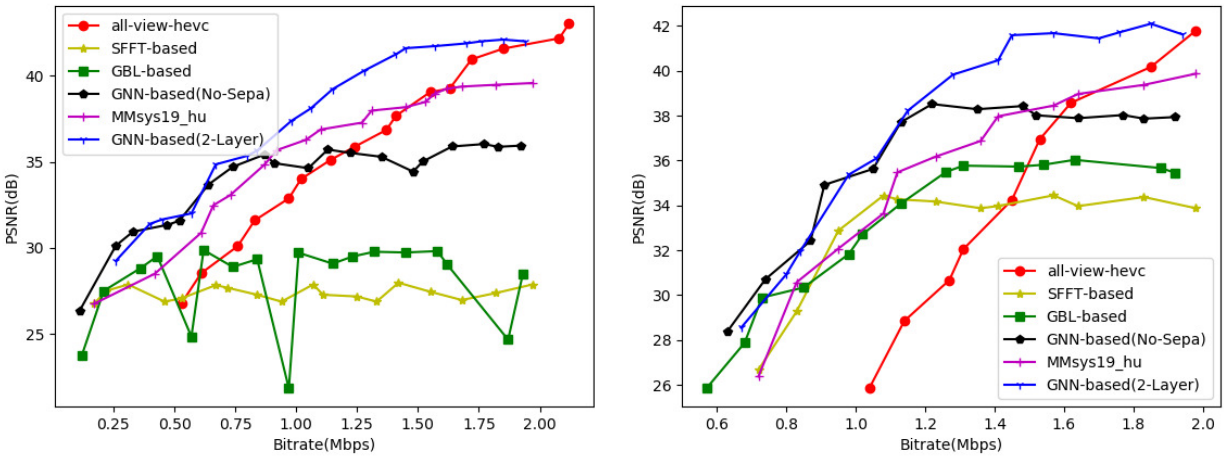


Figure 3.16: Rate-distortion curves for the tested schemes on two datasets: (a) *Stanford Crystal*; (b) *Flower*. (Reprinted from [J5], ©2020 ACM, reused with permission.)

It can also be seen that `MMsys19_hu`, `SFFT-based`, `GBL-based` and `GNN-based (no SEPA)` are superior to `all-view-HEVC` in the case of low bitrate. Thus, the introduction of the LF reconstruction method is very valuable. In particular, even in the absence of frequency separation, marked in the figure as `GNN-based (no SEPA)`, the proposed GNN-based scheme is significantly better than other schemes on all datasets. This confirms the rationale of proposing a reference LF reconstruction algorithm for LF compression. It should be further noted that `GBL-based` has some very low points that do not appear in (b). By analyzing the experimental data, it can be found that this is because GBL uses a gradient search method to determine the graph model, and the results cannot converge when processing complex datasets. This also highlights the reliability of the proposed GNN-based method.

Table 3.2: Parameter values on *Stanford Crystal*

PSNR/dB	α	T_h	Qp_{lp}	Qp_{hp}
29.25	0.74	30	31	39
31.4	0.68	25	33	39
31.65	0.57	17	31	38
32.04	0.52	16	32	37
34.85	0.43	16	31	35
35.35	0.36	15	30	35
35.65	0.31	12	29	36
37.37	0.27	10	29	36
38.1	0.22	10	31	35
39.23	0.18	9	31	34
40.3	0.14	9	33	35
41.25	0.13	7	32	33
41.6	0.14	4	30	35
41.72	0.11	5	31	31
41.88	0.1	2	32	29
42.23	0.09	5	31	29
42.1	0.07	8	33	30
42.36	0.06	4	34	30

Table 3.3: Parameter values on *Flower*

PSNR/dB	α	T_h	Qp_{lp}	Qp_{hp}
28.55	0.93	30	33	36
30.93	0.88	26	31	37
31.95	0.77	22	31	35
35.37	0.72	19	30	35
36.1	0.65	17	32	34
38.22	0.56	18	33	35
39.83	0.42	16	30	33
40.47	0.34	14	32	33
41.59	0.26	12	31	32
41.68	0.17	9	34	33
41.45	0.14	10	30	32
41.72	0.1	8	33	31
42.1	0.12	7	32	29
41.63	0.11	5	32	30

However, the proposed scheme also has limitations. In cases where the image quality of the decoding result is extremely high, the proposed solution cannot meet the demand. This is shown at the end of the red and blue curves in the figure and is mainly due to discarding a portion of the viewpoint data during the encoding process. The information in these viewpoints is almost impossible to fully recover. This is a necessary compromise to achieve high compression ratios.

3.6 Chapter Summary

In this chapter, a two-layer LF compression scheme using reference GNN-based reconstruction is proposed. The scheme can discard some viewpoints before the HEVC encoder and recover them after the HEVC decoder using GNN-based reconstruction. To avoid the influence of high-frequency noise introduced by the reconstruction algorithm, the scheme

separates the high-frequency components and the low-frequency components of the sub-aperture images and independently compresses them. Furthermore, the scheme's dynamic adaptive parameter setting $(\alpha, \mathrm{Qp}_{lp}, T_h, \mathrm{Qp}_{hp})$ gives the proposed scheme the potential to be tuned to meet quality and compression rate requirements of different applications. Compared with several other LF compression schemes that also rely on LF reconstruction, the experimental results show that the proposed scheme is better in terms of compression efficiency and reconstruction quality, both visually and in terms of different evaluation metrics. Especially when dealing with some complex LF images that contain more high-frequency detail information, the gain that the proposed scheme can bring is very considerable.

In summary, the introduction of LF reconstruction in LF compression is a very promising idea that can recover those viewpoints that were discarded at the encoder and make good use of the unique characteristics (the high redundancy of data and the high correlation between viewpoints) of the LF that traditional images do not have.

Chapter 4

Multiple Description Coding for Best-Effort Delivery of Light Field Video Using GNN-based Compression

4.1 Architectural Overview

Figure 4.1 shows the overall architecture, which uses GNN-based LF compression and MDC for delivery in unreliable network environments. As shown by the red dotted box in the figure, the transmitter divides the LF video matrix into descriptions of different downsampling ratios, called levels, in units of viewpoints. For example, it can separate a video with 8x8 angular resolution into three description levels: high, medium and low. The high-level description can be generated by combining subordinate low-level descriptions. To ensure that all descriptions can be independently decoded, a GNN model corresponding to each description is found through training. In each training session, a single description is used as the input, and the original complete LF matrix is used as the label. The transmitter will then use a conventional video coded to transmit only the lowest level descriptions. In addition, the trained GNN models will be transmitted with high reliability.

The blue dotted box in the figure shows the workflow of the receiver. Regardless of whether all basic descriptions are received on time, the receiver will combine the highest-level descriptions as much as possible based on the received data. Finally, the complete LF video data are recovered by running the corresponding GNN model in the forward direction. Since descriptions cannot be arbitrarily combined, a basic description transmission strategy is also designed to make it possible for the receiver to combine higher-level descriptions.

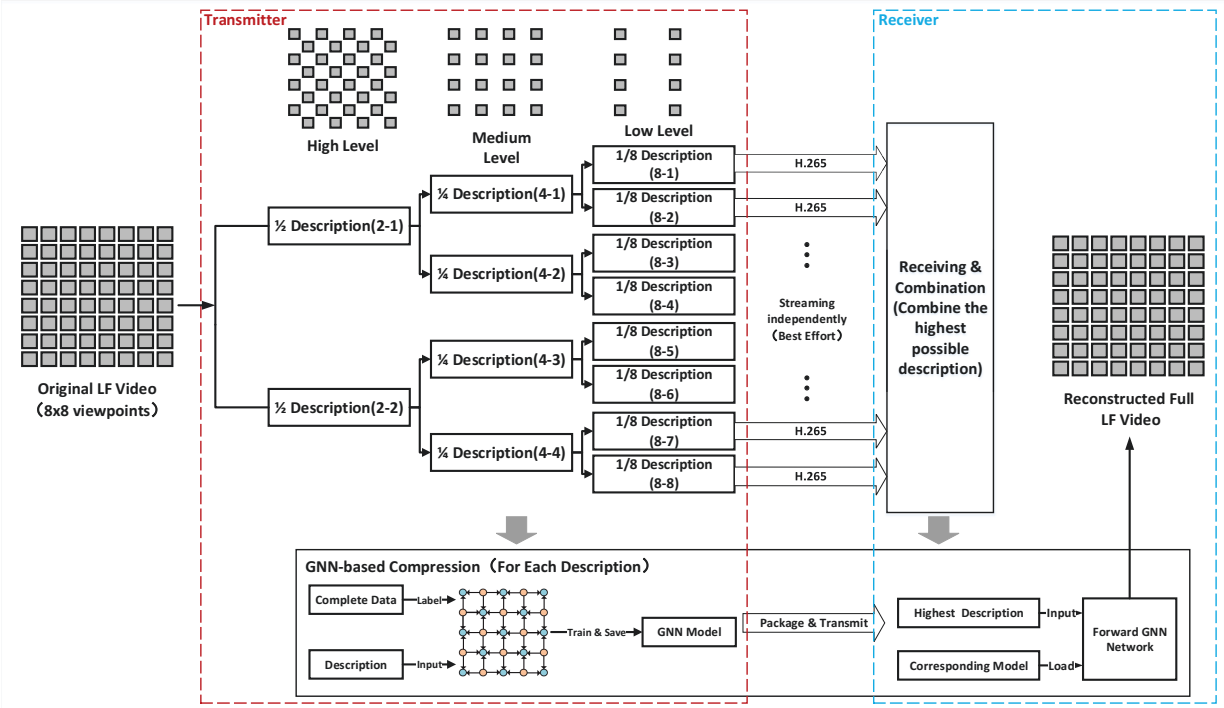


Figure 4.1: MDC scheme for LF video delivery using GNN-based compression. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

4.2 MDC Description Separation Method

Figure 4.1 shows that the description level is determined by the downsampling ratio. The detailed relationship between different levels of description is presented in Figure 4.2. The high-level description downsamples the viewpoint matrix at a 1:2 downsampling ratio, sampling one viewpoint from two adjacent viewpoints. The original LF video can be divided into two high-level descriptions that do not overlap with each other, as shown at the top of Figure 4.2. Similarly, each high-level description can be further divided into two medium-level descriptions, and each medium-level description can be further divided into two low-level descriptions, as shown at the bottom of Figure 4.2. It can also be seen that descriptions cannot be arbitrarily combined. Only two of the low-level descriptions from the same source can be combined into a specific high-level description. For example, description (4-1) can form description (2-1) with description (4-2), but it cannot form a higher-level description with description (4-3). This is mainly because the compression efficiency and quality of the recovered viewpoints will be the best only when uniform

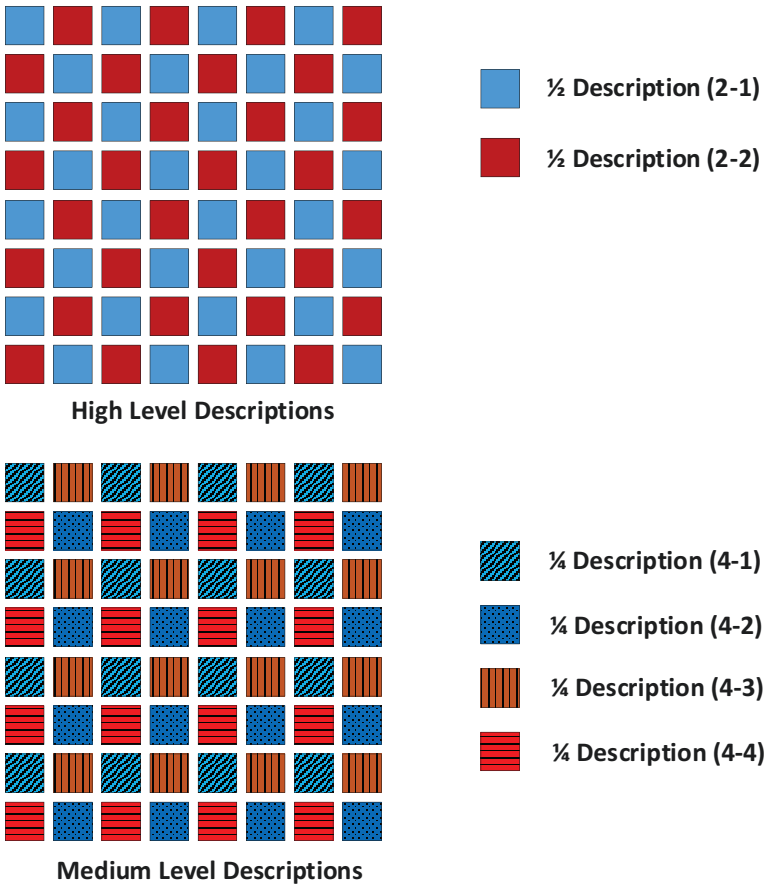


Figure 4.2: Different levels of description. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

downsampled viewpoint sets are used as input. This situation is more obvious when the downsampling ratio of the description is lower.

Figure 4.3 shows the performance difference of GNN compression for medium-level descriptions composed of different combinations of low-level descriptions with a downsampling ratio of 1:8. The upper part is a media description that is evenly composed of two low-level descriptions, while the lower part is a nonuniform combination of low-level descriptions. The average PSNR of the viewpoint stream is used to evaluate the quality of the recovered LF video. The results show that the recovery quality of the uniform combination is better than that of the nonuniform combination. In fact, the PSNR of the recovery results using only one single low-level description can reach 29 dB. Therefore, combining the

low-level descriptions from different sources yields minimal improvement in terms of the final recovery result. The proposed scheme does not consider the description combination from different sources. This also avoids generating too much data for the GNN model, which helps to reduce both networking and processing resources.

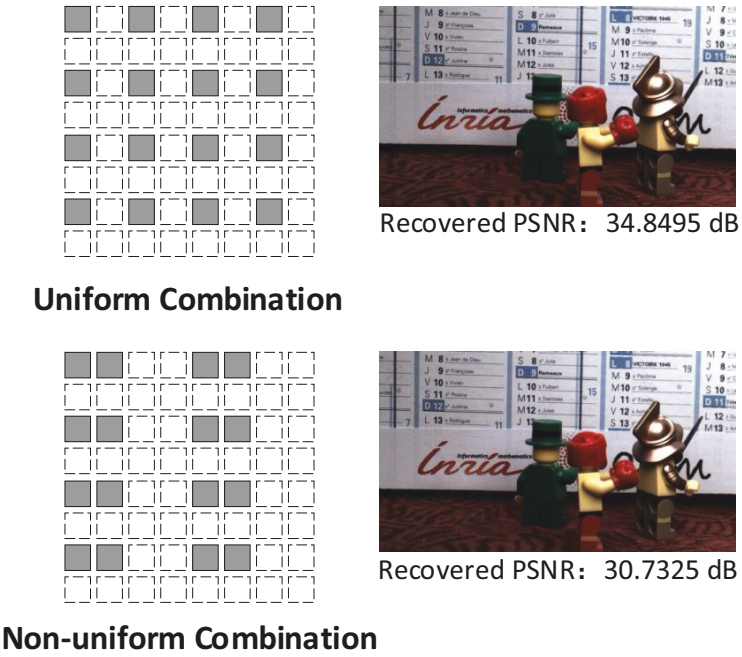


Figure 4.3: Effect of uniform and nonuniform description combinations on GNN recovery results. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

4.3 GNN Model for LF Description Compression

As introduced in previous chapters, one can use the vertices and edges of the graph to efficiently represent different viewpoints and their correlation due to the special data structure of the LF matrix. By taking the low downsampling ratio description as the input and the complete original matrix as the label, a GNN model can be trained, as shown in Figure 4.4. During training, the GNN can iteratively update the hidden state of all nodes until the network model converges, which ensures that each node can fully perceive other nodes on the graph and that all viewpoints can be precisely reconstructed through the graph model.

With the help of the GNN, the complete LF matrix can be independently reconstructed using the description. There are two methods to control the quality of the GNN reconstruction results. The first method, which was used in the last chapter, is to control the quality of the input viewpoint without changing the network structure. After training the GNN model, for those viewpoints that are reserved for further HEVC encoding and transmission, a lower bitrate encoding configuration is used to achieve a higher compression ratio. However, at the decoding end, the degradation of the input viewpoint quality will lead to poorer GNN reconstruction output results. The second method is to reduce the number of input viewpoints. When training the network, the interchange between the decoding quality and compression ratio can be realized by using different numbers of viewpoints as input. However, a more complex and changeable GNN network structure is needed for support.

To achieve the scalability of LF MDC, the second quality control method is adopted in this work. As shown in Figure 4.4, the GNN network structure is divided into different basic cells. The numbers of nodes and edges in the basic cell are affected by the downsampling ratio of the input description. The dark-colored nodes represent the viewpoint as input, and the light-colored nodes are the viewpoints to be generated. All edges only exist inside the basic cell, and their directions are from the input node to the node to be generated. Basic cells are independent of each other, and no interdependence is established between the nodes to be generated. When the number of viewpoints used as input is the same, the network convergence results will be better when these viewpoints are uniformly spaced, as shown in Figure 4.3. This is mainly because when the input nodes are more evenly distributed in the basic cell, the average distance between the nodes to be generated and the input nodes are closer. A closer distance means that the higher the relevance between the nodes is, the easier the prediction. When the input nodes are unevenly distributed, some nodes will be generated too far away from the reference node; thus, it is more difficult for these nodes to converge during the training process. In addition, the uniform description structure is more convenient for the algorithm to adaptively generate the corresponding network structure according to the downsampling ratio of the description.

It should be noted that due to LF sequence variation in the depth distribution of targets (objects, pedestrians, vehicles, etc.), the intervals between the lenses of LF capture devices are not the same and range from a few millimeters to a few centimeters. Consequently, the viewpoint disparity between LF sequences greatly differs from a few to dozens of pixels. Therefore, in LF encoding, the tradeoff between the accuracy and the generalization of the GNN model has become a crucial issue. Whether to use all sequences to train an "average" network model or to generate a "custom" model for each sequence has results in two different design approaches. The advantage of the former design approach is that after

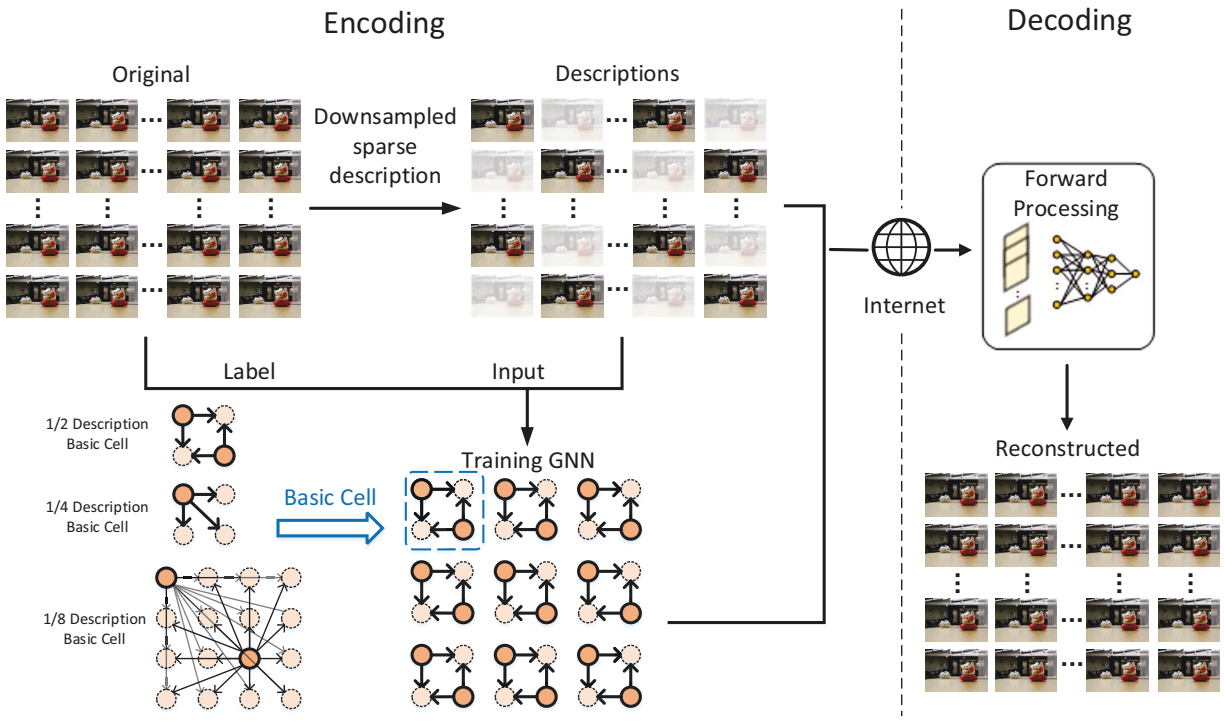


Figure 4.4: GNN model for LF description compression. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

universal training, videos no longer require additional training; however, the reconstruction quality of all sequences is sacrificed. As a result, the decoded result for each sequence is no longer controllable or predictable. Thus, the latter design approach, that is, the design approach in which the GNN models used by different sequences are not uniform, is adopted in this work. The generalization of the GNN model is not considered in this work.

4.4 The Best-effort Scheduling Algorithm

Since low-level descriptions cannot be randomly combined into high-level descriptions, it is necessary to properly design the transmission sequence of the basic description. This can effectively reduce the impact of random network packet loss or overtime retransmission on the LF video delivery process and increase the possibility of the receiver combining a higher-level description from the received data in real time.

In the proposed scheme, only the lowest level descriptions, which are defined as the basic description, are transmitted. Each higher-level description needs to be composed of two lower-level descriptions. Let us assume that the total number of descriptions to be transmitted is N, the data size of each basic description is k megabytes, and the total available bandwidth is B Mbps when the network condition is stable. As shown in Figure 4.5, the transmission process is cut into time segments of length t; The segment duration t can be set arbitrarily, e.g. setting t as the period of actual network fluctuations. In this chapter, an optional choice of t is provided according to the average burst loss length derived from the Gilbert model. The i-th basic description transmitted is defined as D_i.

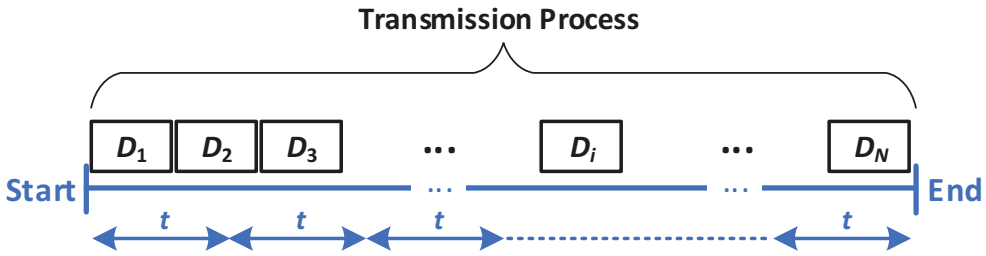


Figure 4.5: Transmission process for the MDC descriptions of the LF video. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

The transmission start and end times of D_i are shown in Equation (4.1).

$$\begin{aligned} T_{i-start} &= \frac{8 \times (i-1) \times k}{B} \\ T_{i-end} &= \frac{8 \times i \times k}{B} \end{aligned} \tag{4.1}$$

For a lossy network environment, the Gilbert model, which is a special case of the Gilbert–Elliot model, is used to simulate packet loss. The Gilbert–Elliot model is a Markov model that consists of two states of the network, the good (G) and bad (B) states. The packet loss rate when the network is in the B state will be higher than that in the G state. The probability of the network state changing from the G state to the B state is α, and vice versa is β. The Gilbert–Elliot model is widely used in the simulation of burst-noise transmission channels [63]. When the packet loss rate in the G mode is set to 0% and the packet loss rate in the B mode is set to 100%, then the Gilbert–Elliot model is simplified to the Gilbert model [64]. The transition matrix of the Gilbert model can be given as

Equation (4.2).

$$P = \begin{pmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{pmatrix} \tag{4.2}$$

The probability distribution of the stationary state of Equation (4.2) can be calculated as Equation (4.3):

$$\begin{aligned} \pi_G &= \frac{\beta}{\alpha+\beta} \\ \pi_B &= \frac{\alpha}{\alpha+\beta} \end{aligned} \tag{4.3}$$

where π_G and π_B are the probabilities of the G and B states, respectively.

Because the packet loss rates of the G and B states are 0 and 1, respectively, the overall packet loss rate p can be calculated by the conditional probability, as shown in Equation (4.4).

$$\begin{aligned} p &= \pi_G \times 0 + \pi_B \times 1 \\ &= \frac{\alpha}{\alpha+\beta} \end{aligned} \tag{4.4}$$

In the actual network, in addition to the packet loss rate, the average burst loss length is a very important factor because under the same packet loss rate, the distribution of lost packets may occur intensively and continuously or scatteredly and discretely. These two packet loss situations will have different impacts on the network service, so the impact of the average burst loss length also needs to be considered when designing the scheduling algorithm. The probability of the burst loss length of the Gilbert model can be expressed by Equation (4.5).

$$P_{length}(n) = (1-\beta)^{(n-1)} \times \beta \quad 1 \leq n \leq M \tag{4.5}$$

where $P_{length}(n)$ represents the probability of losing n consecutive packets. M is the maximum number of packets. Therefore, the expectation of n is the average burst loss length. The expectation E can be formulated as Equation (4.6):

$$E = \frac{(1-(1-\beta)^M)}{\beta} - M(1-\beta)^M \tag{4.6}$$

When $\beta \in [0,1]$ and $M \to +\infty$, the average burst loss length E can be simplified to:

$$E = \frac{1}{\beta} \tag{4.7}$$

It is conceivable that part of the description may be at the boundary of two time segments, as shown in Figure 4.6. Compared to the description of the nonboundary area, the description of the boundary area is riskier.

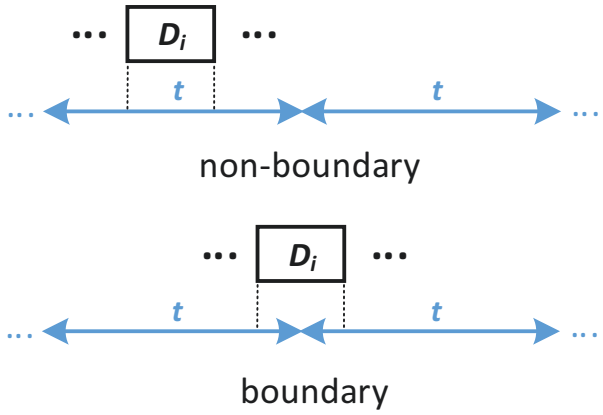


Figure 4.6: The description inside a time segment or on the boundary of two time segments. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

The probability that description D_i cannot be completely transmitted to the receiver on time is shown in Equation (4.8) and Equation (4.9). When the average burst loss length E is small, t is small, which means that short-term packet loss frequently occurs on the network; this will have a more adverse effect on the transmission of descriptions.

$$t = \frac{E \times Packetsize}{B} \tag{4.8}$$

$$P_i = \begin{cases} P_{length}(E) & (T_{i-start} \bmod t) = (T_{i-end} \bmod t) \\ P_{length}(E)^2 & (T_{i-start} \bmod t) \neq (T_{i-end} \bmod t) \end{cases} \tag{4.9}$$

Although there will be multiple descriptions at the same level, one of them will be chosen in the scheduling algorithm, and the basic description of its subordinates will be given priority. As shown in Figure 4.1, first, the basic description of the subordinates of the high-level description (2-1), which refers to descriptions with serial numbers (8-1) to (8-4), will be handled. The scheduling algorithm, as shown in Algorithm 4.1, will arrange the optimal sending order for each basic description according to their serial numbers.

Initially, in the proposed scheduling algorithm, the first basic description is placed at the top of the sending sequence S. This is because the schedule of the following descriptions

Algorithm 4.1 Scheduling Algorithm for Description Delivery.

Input:

Set of available sending order A and number of basic descriptions N.

Output:

Sending sequence S

initial $A \leftarrow [2:N]$, $S \leftarrow$ `zeros`(N) and $S_1 = 1$
for $i \in [2:N]$ **do**
  for $j \in [1:N-1]$ **do**
    if $S_{i-1}+j \in A$ && `IsSameSegment`$(S_{i-1}+j, S_{i-1})$ is **False then**
      if `IsBoundary`$(S_{i-1}+j)$ is **False then**
        $S_i = S_{i-1}+j$, **break**
      end if
    end if
    if $S_{i-1}-j \in A$ && `IsSameSegment`$(S_{i-1}-j, S_{i-1})$ is **False then**
      if `IsBoundary`$(S_{i-1}-j)$ is **False then**
        $S_i = S_{i-1}-j$, **break**
      end if
    end if
  end for
  if $S_i == 0$ **then**
    $S_i \leftarrow$ `Search_without_Bound` (S_{i-1}, A)
  end if
  if $S_i == 0$ **then**
    $S_i \leftarrow$ `Search_without_Bound&Adjacency` (S_{i-1}, A)
  end if
  Remove S_i from A
end for

needs to consider the order of the previous descriptions. Thus, an initialization seed should be defined. The search strategy starts from the position of the previous description and searches for both sides of the sending sequence. The advantage of this method is that basic descriptions from the same source can be sent out as close as possible so that the receiver can receive enough data to quickly reassemble the high-level description without waiting for the LF video to be completely received. The search process has two constraints:

1. The current description and the previous description cannot be sent in the same time segment.

2. The current description on the border of two adjacent time segments cannot be sent.

The first constraint prevents two consecutive descriptions that are considered to be too close to one another from being sent; otherwise, both descriptions will be lost if a network failure occurs, reducing the possibility that the receiver can combine high-level descriptions. The second constraint is set according to Equation (4.9). It reduces the delivery risk of the description with the highest scheduling order and ensures the reliability of the basic description transmission under the same source of high-level description. The scheduling algorithm will use the above constraints and perform up to three scheduling searches for the current description. If the first search fails to find a suitable position, the second constraint will be removed in the second search. If the second search fails, all constraints will be released, and the algorithm will find the nearest available position in the sending sequence. Next, a practical example is used to show the process of the proposed scheduling algorithm.

One LF sequence is used as an example for the detailed presentation. The example sequence has 8×8 viewpoint streams, and the downsampling ratio of its basic description is 1:16. There are a total of 16 basic descriptions, which are sequentially numbered (16-1) to (16-16) according to the structure shown in Figure 4.1. The playback frame rate of the sequence is set as 24 FPS, and 150 consecutive frames are regarded as one window. This means that the LF video will not rebuffer if all the description data of the next window are cached before the previous window finishes playback. In each window, the data size of each basic description after H.265 encoding is approximately 5.6 MB. It is easy to calculate the bandwidth required to transmit all descriptions in real time, which is approximately 14.3 Mbps. To make the experimental results more significant, the time segment is set to 0.5 seconds, which indicates that there is a possibility of transmission errors in the network every half second.

Figure 4.7 shows the composition structure of the description and the optimal sending sequence given by the proposed scheduling algorithm. According to Algorithm 4.1, D_1

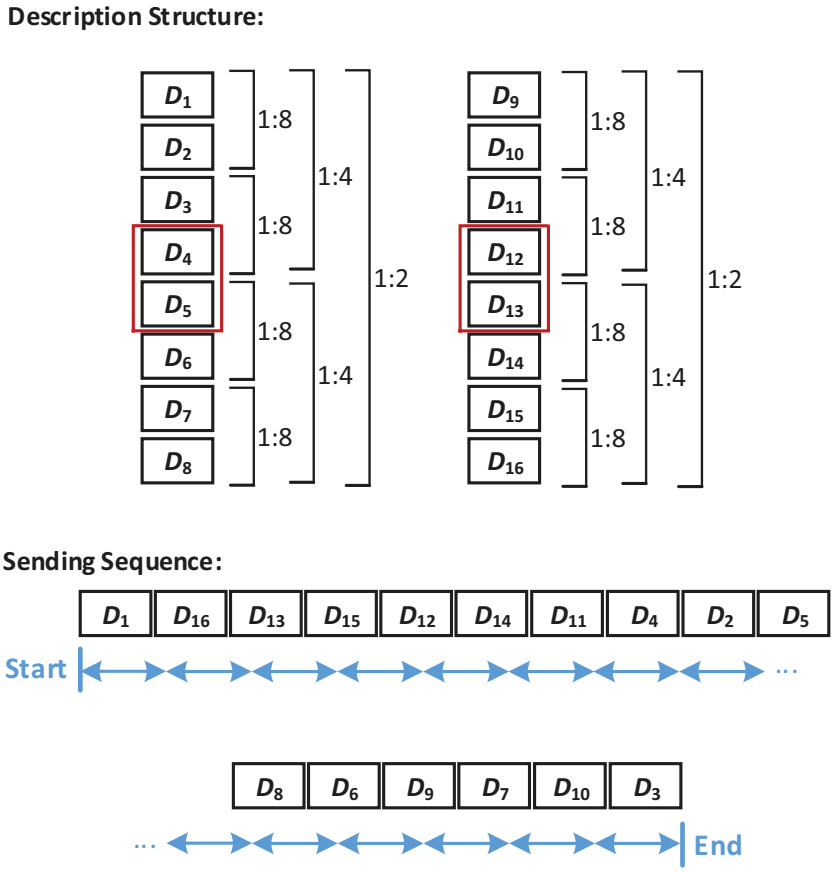


Figure 4.7: Description structure and sending sequence of the proposed scheduling algorithm. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

is the first description and is sent by default. For D_2, except for order 1, the sending order from 2 to 16 is optional ($A = [2{:}16]$). Therefore, the scheduling algorithm searches for the ideal sending position for D_2 starting from order 2. However, if D_2 is sent in order 2, part of it will be sent in the same segment as D_1, and the **IsSameSegment** function (corresponding to the first constraint) will be true. Therefore, order 2 is not a position that meets the constraint requirements, and the algorithm continues to search backward. Although orders 3 to 8 meet the first constraint, they will cause the transmission of D_2 to span two segments, which does not meet the second constraint. Therefore, the **IsBoundary** function will be true, the two constraints are not met at the same time, and the search will continue. When order 9 is reached, all constraints are met, so D_2 will be sent in the ninth position. Thus, S_2 is set to 9, and 9 is removed from A. For D_4, because D_3 has

been scheduled to be sent at the end of the queue, the algorithm performs a forward search after searching backward with no success. For descriptions with a larger index, since it is impossible to find a position that fully meets the two constraints when it is their turn, the restrictions are released one by one to complete the subsequent search process, as shown by the functions **Search_without_Bound** and **Search_without_Bound&Adjacency**.

D_4, D_5, D_{12} and D_{13}, which are shown in the red dotted boxes, are the most important basic descriptions because, compared to other basic descriptions, they are necessary elements to compose a high-level description (1:4 or 1:2). However, if they are sent in the order of the original numbers, they will be in the same time segment, which increases the risk of losing them due to network impairments. It can be seen that the sending sequence provided by the proposed scheduling algorithm can avoid these risks well. Real-time performance is sacrificed in pursuit of reliability in the transmission of D_1-D_8, which may result in a longer wait time for D_1-D_8 to be completely received and form a high-level description under a good network environment. Compared with the orders above, the orders assigned to D_9-D_{16} will be consecutively adjacent (both reliability constraints have been released) and placed in the front part of the queue (the algorithm will first search for suitable positions in the backward direction). However, the transmission of D_9-D_{16} can compensate for the lack of real-time performance. For example, when the speeds of receiving and decoding cannot keep up with the speed of playback (or processing), the decoder does not have to wait for all the data to be received; it only needs to receive the first 6 basic descriptions to decode a 1:4 high-level description with D_{13}-D_{16}. Through GNN reconstruction, the 1:4 high-level description can recover at least a complete light field matrix of more than 35 dB. This quality of decoding output can meet the minimum requirements of most light field processing and applications according to the author's previous work [J4]. The design of the proposed scheduling algorithm is very conducive to flexibly adjusting the decoding strategy to control the buffer when the decoder faces the risk of a rebuffer. A balance between the real-time performance and reliability of LF video transmission can be achieved using this algorithm.

As deduced from Algorithm 4.1, the computational complexity of the proposed scheduling algorithm is $O(N^2)$, where N is the total number of descriptions that need to be sent at the current moment. When N is 16, the average time consumed by the scheduling algorithm is 0.22 milliseconds. LF streams with 16 descriptions and an FPS that does not exceed 4545 ($\approx 1000/0.22$) can be scheduled in real time, far exceeding actual FPS demand. Therefore, the additional coding overhead caused by the scheduling algorithm is negligible.

4.5 Experimental Evaluation

4.5.1 Experimental Setup

The test dataset contains seven LF video sequences that have been published on the internet. The open-source datasets of [65] and [66] are combined to construct the LF video test set. Wang et al. [65] developed a hybrid imaging system that can synthesize high FPS LF video. Given a 3 FPS light field sequence and a standard 30 FPS 2D video, the system can generate a full light field video at 30 FPS. Guillo et al. [66] presented a dataset of LF video captured by an R8 Raytrix camera. The sequences *cats*, *dance*, *train1* and *train2* are from [65], and the sequences *boxer*, *chess1* and *chess2* are from [66]. Detailed information and samples of all LF sequences are shown in Table 4.1 and Figure 4.8.

Table 4.1: Information on the tested LF video sequence.

Sequence	Spatial Res.	Angular Res.	No. of Frames
cats	512×352	8×8	109
dance	512×352	8×8	80
train1	512×352	8×8	84
train2	544×320	8×8	97
chess1	960×540	5×5	100
chess2	960×540	5×5	101
boxer	1920×1080	5×5	101

To verify the effectiveness of the proposed scheme, simulation experiments with real network trace records are conducted. The mobile network trace dataset from [2, 3] and the raw traffic data of US internet users published by the FCC (Federal Communications Commission) in the first two quarters of 2020 are used. [2] and [3] mainly collected real-time bandwidth traces for 4G/LTE networks when the tester rides in different vehicles, such as a bus, train, car and bicycle. As shown in Figure 4.9, on these moving vehicles, the network bandwidth greatly fluctuates, which is detrimental to the transmission of LF video with high and stable bandwidth requirements. The FCC data are indexed according to the target URL accessed by the users' app, and all kinds of network traffic are counted together (including wired networks, WiFi, and cellular networks). The statistics on the bandwidth of video apps, such as YouTube, Hulu and Netflix, are extracted, and the real-time throughput of the users' video streaming services is calculated. All network trace

Figure 4.8: Examples from the tested LF video sequence. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

datasets used in this work have been uniformly converted to Mahimahi [67] format before further processing.

The proposed scheme was run on Python 3.61 using the HM encoder for stream encoding. DGL and PyTorch were used to implement the GNN. The scalability extension of HEVC was used as a baseline to compare with the proposed scheme. In addition to SHEVC, various MDC-based schemes proposed or referenced by [68] were also used as comparison baselines.

4.5.2 Experimental Results

In the subsections below, the results of the extensive experiments in terms of recovery quality, compression performance, efficiency of the scheduling algorithm, and adaptability to sudden network fluctuations and network instability will be shown first.

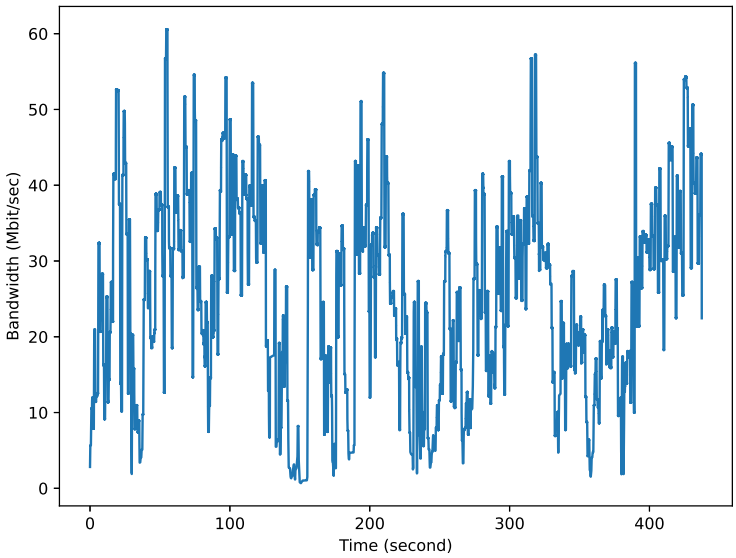


Figure 4.9: Bandwidth fluctuations using the network trace dataset in [2] and [3]. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

4.5.2.1 Recovery Quality

In this section, the quality of the recovered LF video with different levels of description is first shown. The sequence *cats* is cut into 4 levels of description. The downsampling ratio of the lowest level basic description is 1:16. Because the recovery results of the viewpoints included in the input description will be better than other viewpoints, the recovery quality of viewpoints is different. Figure 4.10 shows the distribution of PSNR for all viewpoint restorations with different levels of description on *cats*. Only one random description for each level is selected. Table 4.2 lists the mean and standard deviation of the PSNR distribution shown in Figure 4.10. To provide more comparisons of subjective and objective quality assessment metrics, the results of SSIM and Netflix's VMAF [61] are also given in Table 4.2.

It can be seen from the results on *cats* that the average PSNR result of the high-level description is much better and the standard deviation is also relatively smaller, which means that although low-level descriptions require fewer viewpoints, their recovery results are worse and more unstable. Figure 4.11 shows the recovery PSNR distribution of 1:4 level descriptions for 10 consecutive frames. From $F1$ to $F10$, each box represents the quality of the reconstructed viewpoint of one frame. Over time, the recovery quality of the description at the same level is very stable, and the mean values and standard deviations

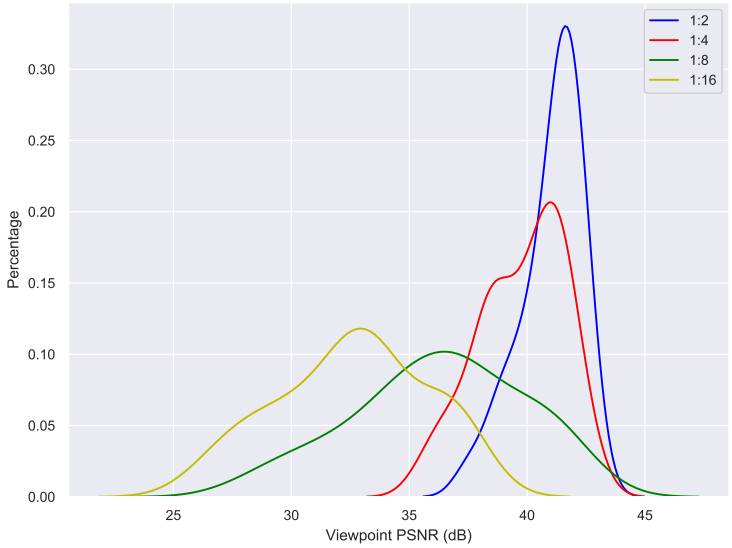


Figure 4.10: PSNR distribution of the recovery results for all viewpoints on *cats*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

Table 4.2: Mean and standard deviation of the recovered PSNR for different level descriptions on *cats*.

Sampling Ration	Mean PSNR/dB (std)	Mean SSIM (std)	Mean VMAF (std)
1:2	41.0232 (1.2739)	0.9865 (0.0038)	95.2142 (1.5327)
1:4	39.7220 (1.7764)	0.9762 (0.0073)	91.7359 (2.0832)
1:8	36.1864 (3.5778)	0.9587 (0.0250)	86.1722 (2.7348)
1:16	32.5674 (3.1137)	0.9399 (0.0303)	87.0089 (3.6206)

of the results are almost completely unchanged. The test results of the remaining six LF video sequences are shown in Figure 4.12 to Figure 4.17. The blue, red, green, and brown curves in these figures correspond to the recovered viewpoint's PSNR distribution of the descriptions with sampling rates of 1:2, 1:4, 1:8, and 1:16, respectively. The experimental results prove that the MDC scheme designed in this chapter can always recover all the viewpoint streams of the LF video, and the higher the integrity of the received data, the better the recovery quality.

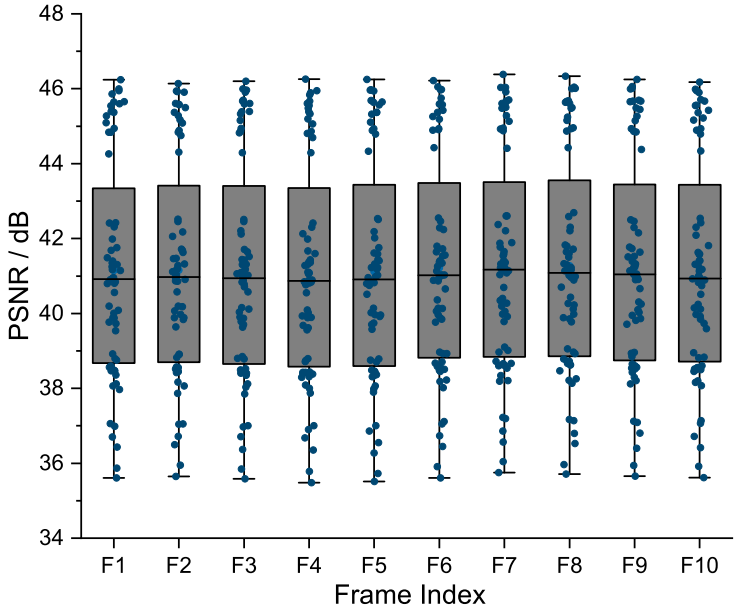


Figure 4.11: Recovery PSNR distribution of 1:4 level descriptions for 10 consecutive frames. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

4.5.2.2 Compression Performance

Here, we show the compression performance of the proposed scheme. Figure 4.18 to Figure 4.24 show the rate-distortion curves of all schemes on seven test LF video sequences. The proposed method is compared with the SHEVC method and MDC methods using randomly offset quantizers [69], two-description scalar quantization [70], a JPEG2000-compatible multiple description scheme [71], an optimized multiple description [72] and a convolutional neural network [68], which are labeled as "SHEVC", "MDROQ", "MDSQ", "Tillo's", "Bai's" and "MDCNN", respectively.

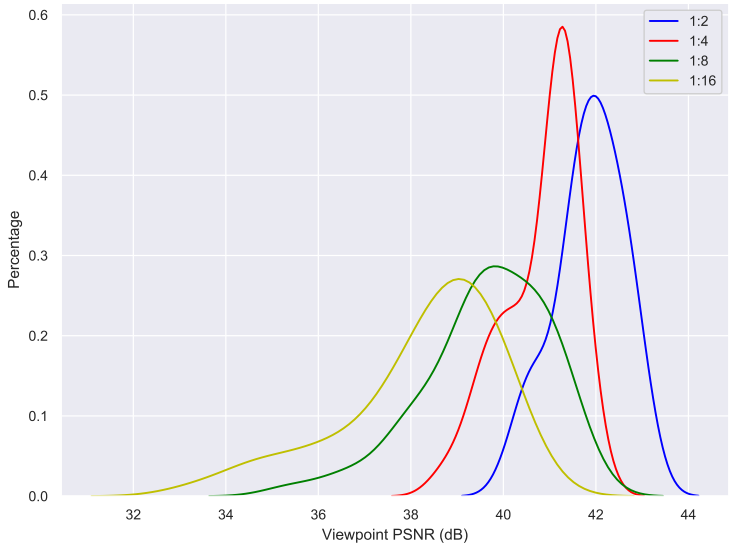


Figure 4.12: PSNR distribution of the recovery results on *dance*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

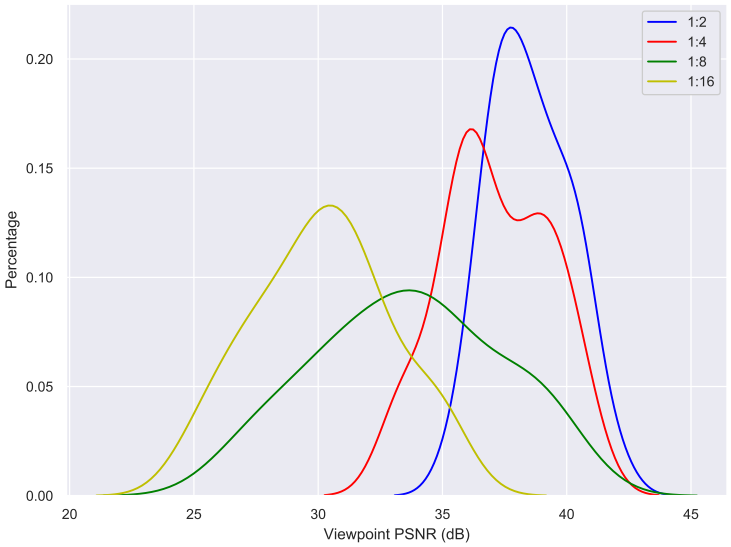


Figure 4.13: PSNR distribution of the recovery results on *train1*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

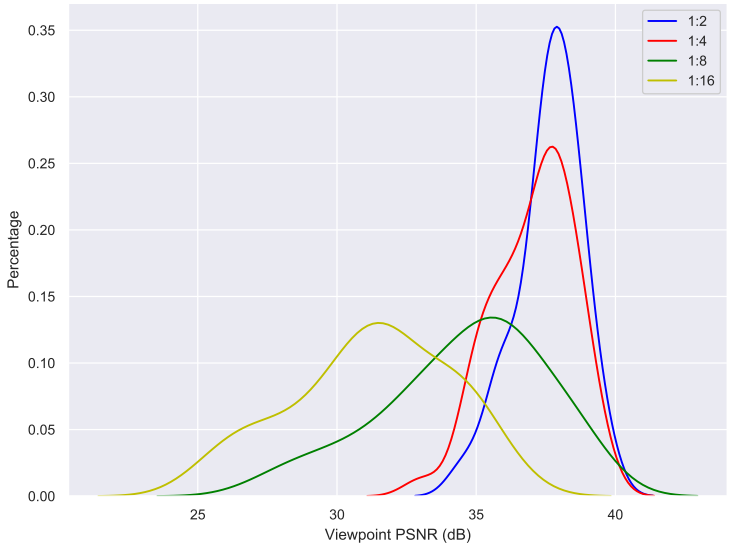


Figure 4.14: PSNR distribution of the recovery results on *train2*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

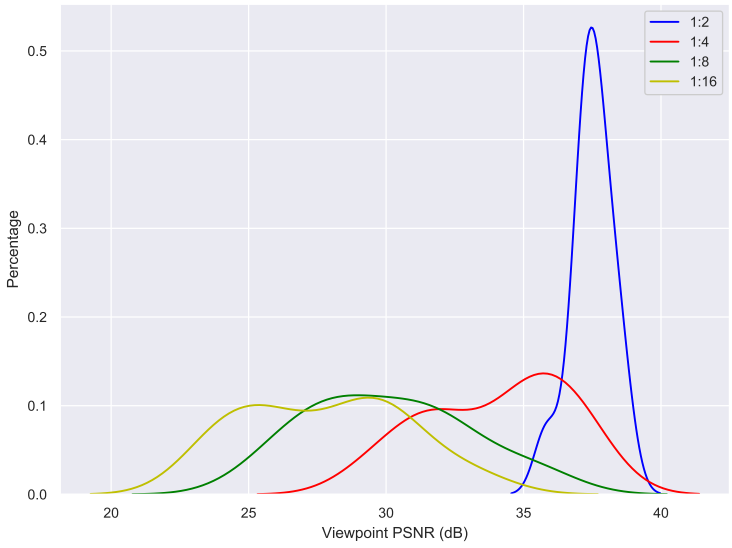


Figure 4.15: PSNR distribution of the recovery results on *boxer*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

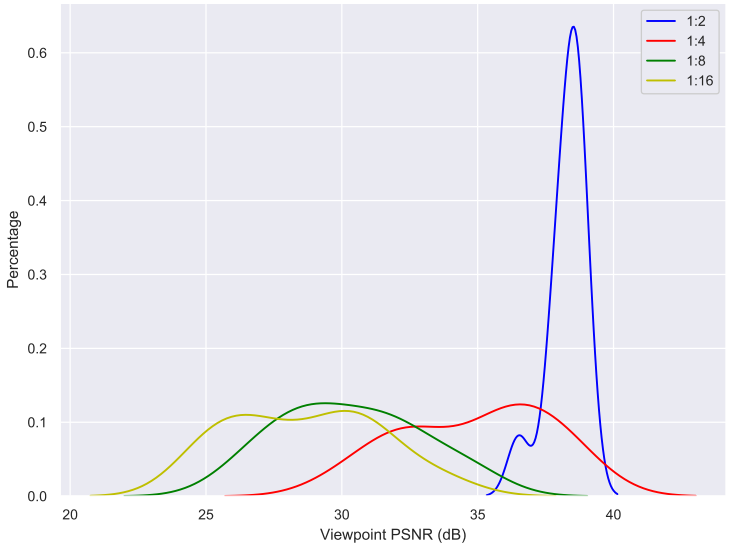


Figure 4.16: PSNR distribution of recovery results on *chess1*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

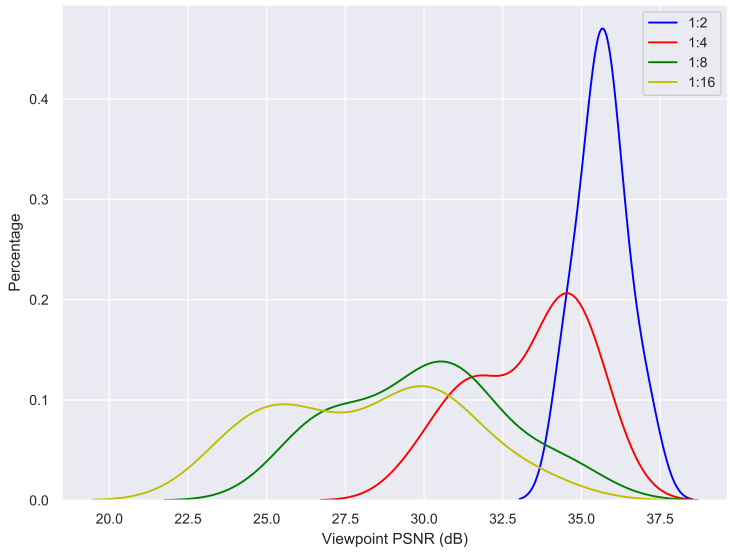


Figure 4.17: PSNR distribution of recovery results on *chess2*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

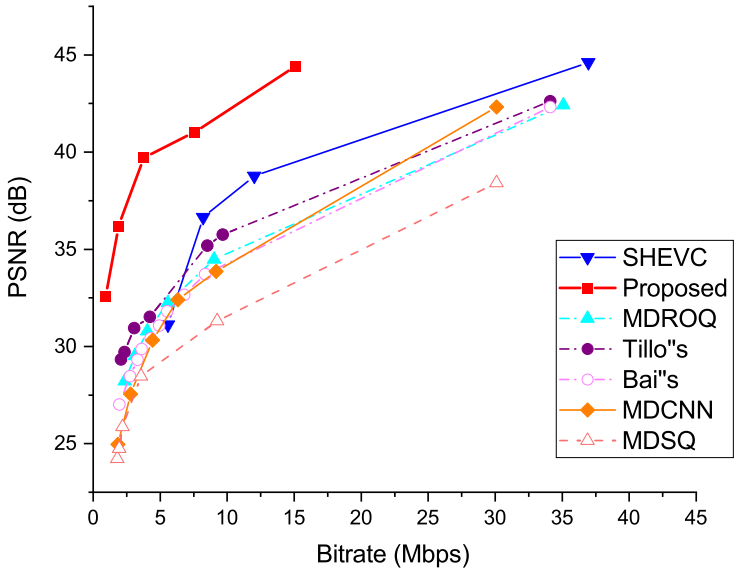


Figure 4.18: Rate-distortion curves of all schemes on *cats*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

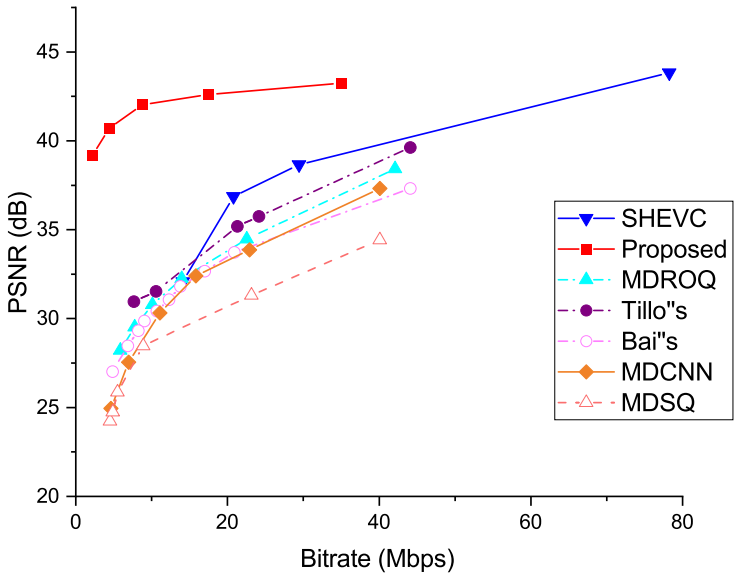


Figure 4.19: Rate-distortion curves of all schemes on *dance*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

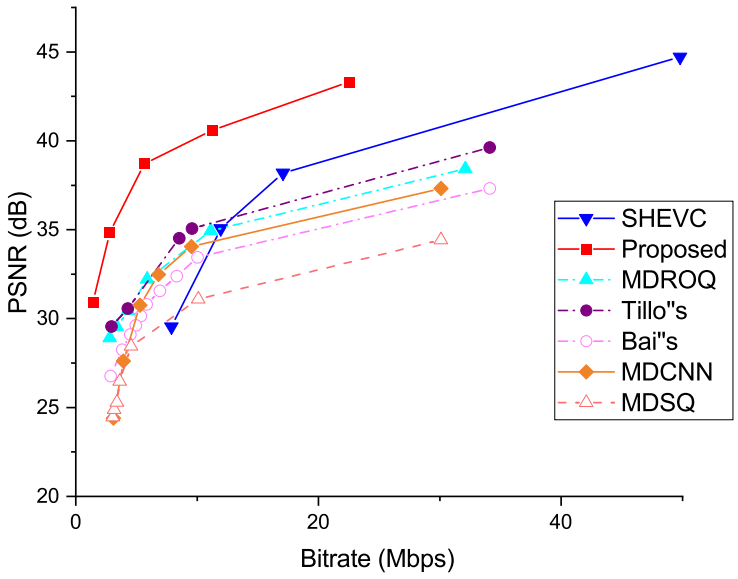


Figure 4.20: Rate-distortion curves of all schemes on *train1*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

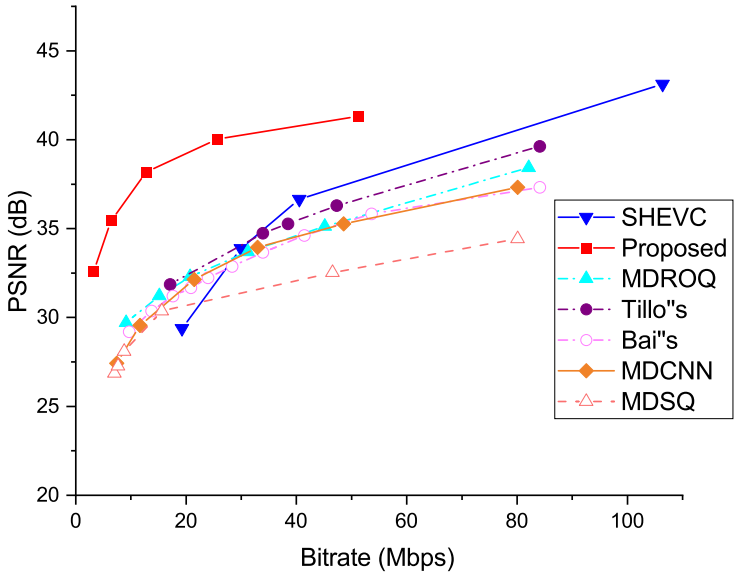


Figure 4.21: Rate-distortion curves of all schemes on *train2*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

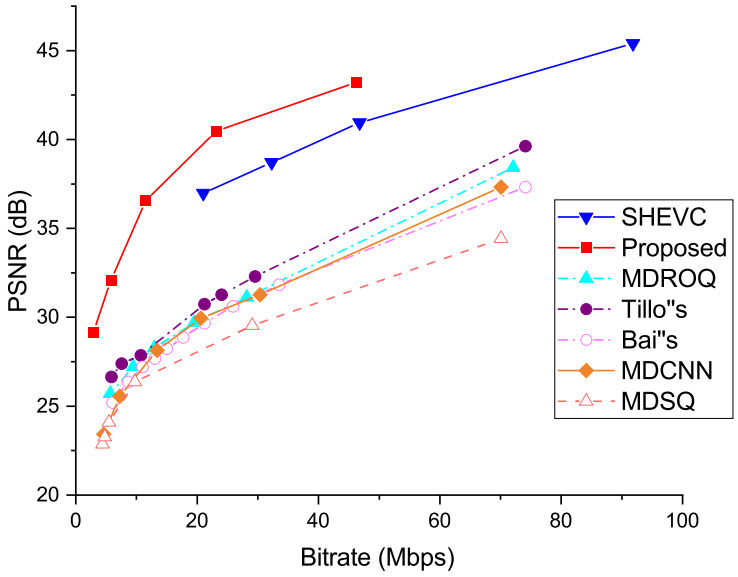


Figure 4.22: Rate-distortion curves of all schemes on *boxer*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

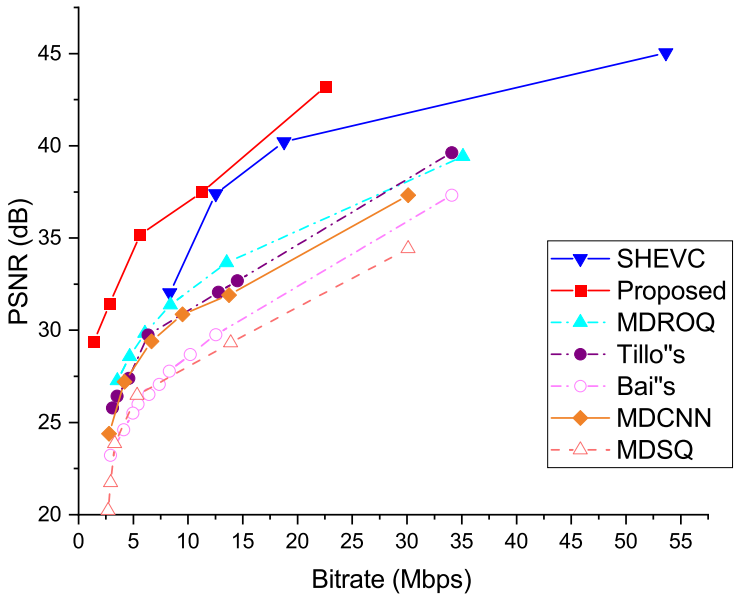


Figure 4.23: Rate-distortion curves of all schemes on *chess1*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

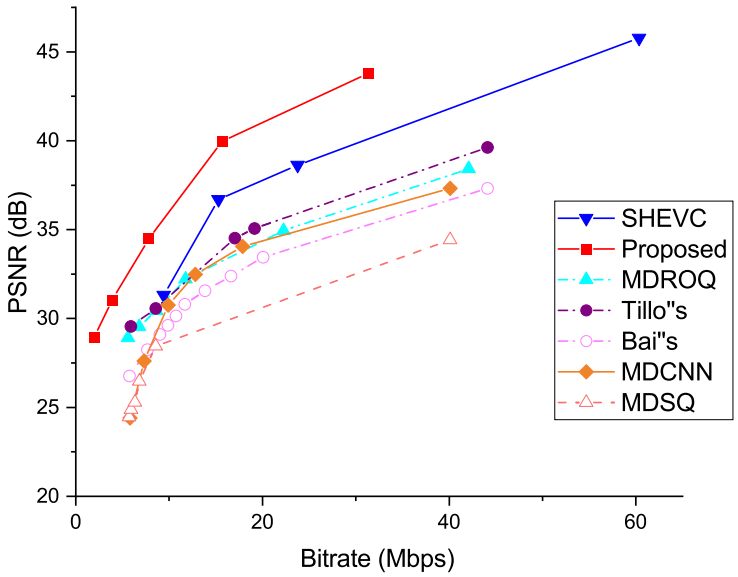


Figure 4.24: Rate-distortion curves of all schemes on *chess2*. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

It should be noted that since all baseline schemes are designed for traditional videos, they cannot be directly used to encode LF video. Therefore, when testing these schemes, LF video is treated as multiple video sequences for independent encoding in units of viewpoints. The results of SHEVC, MDROQ and MDCNN were obtained by using the software repository or source codes provided by the authors. The MDSQ scheme was reproduced on MATLAB according to the instructions provided by [70].

As seen from Figure 4.18 to Figure 4.24, since the proposed scheme is specifically designed for the correlation between LF video viewpoints, it is far superior to other schemes in the compression performance of LF video. It also needs to be noted that in the case of a high bitrate, the compression performance of SHEVC is also better than that of the other baseline schemes. This is mainly because most MDC-based schemes usually use their own custom quantization methods when generating descriptions; therefore, HEVC encoders cannot perform very well when processing descriptions with special quantization ranges. Because SHEVC is developed based on the expansion of HEVC, it successfully inherits the excellent compression performance of HEVC. Taking this point into consideration, the proposed scheme does not specifically perform pixel quantization but directly stores different descriptions in the form of YUV420 to maximize the compatibility with different existing video and image encoders. The proposed scheme can make good use of the excellent

performance of traditional coding schemes in the spatial domain of LF video and at the same time further improve the compression performance in the angular domain with the help of the GNN-based LF reconstruction method. In addition, there is another reason for the large compression efficiency difference between the proposed method and SHEVC. SHEVC introduces a large amount of redundant data to ensure the realization of scalability. Even if directly compared with HEVC, the bitrate required by SHEVC is significantly higher than that by HEVC. Compared with these methods, the proposed method only needs a small amount of extra data to transfer the GNN model file. These GNN model files are only tens of KB and can be continuously used. Therefore, the gain in compression efficiency of the proposed scheme is significant.

4.5.2.3 Efficiency of the Scheduling Algorithm

Here, we prove that the scheduling algorithm can effectively help the proposed scheme better adapt to networks with moderate or high packet loss rates. In this work, the packet loss rate according to PING instruction is determined, i.e., packets with a size of 32K bytes are continuously sent to an IP address, and the packet loss rate is calculated by counting the number of lost packets. Generally, if the packet loss rate of a network in a city exceeds 1%, it means that the network condition is very poor; network congestion may be severe or the physical connectivity of the core node may be very unstable. However, considering that the LF camera may be used as the front-end acquisition system of many visual 3D modeling platforms in the future, the performance of the network in rural areas, where wireless communication is unstable, is also very important. Therefore, extreme network environments with packet loss rates of 3% and 5% are also considered in the tests. At the same time, different average burst loss lengths are also used in the experiment to study their impact on LF streaming.

A simulation of LF video streaming is performed on all network bandwidth traces of the test dataset. During the streaming process, seven LF video sequences were continuously and cyclically coded and transmitted. Figure 4.25 to Figure 4.27 show the impact of the packet loss rate on the scheduling algorithm under different average burst loss lengths. Each curve corresponds to the output quality distribution of frames under different conditions. Under the premise of the same average burst loss length, each subfigure uses 1%, 3% and 5% packet loss rates to obtain three pairs of cumulative distribution curves. The blue lines are the result of sending the descriptions in a random order when the scheduling algorithm is not applicable. The red lines are the cumulative distribution function (CDF) of all recovered frames' PSNR after using the scheduling algorithm.

Figure 4.25 to Figure 4.27 show that in the case of various packet loss rates and average

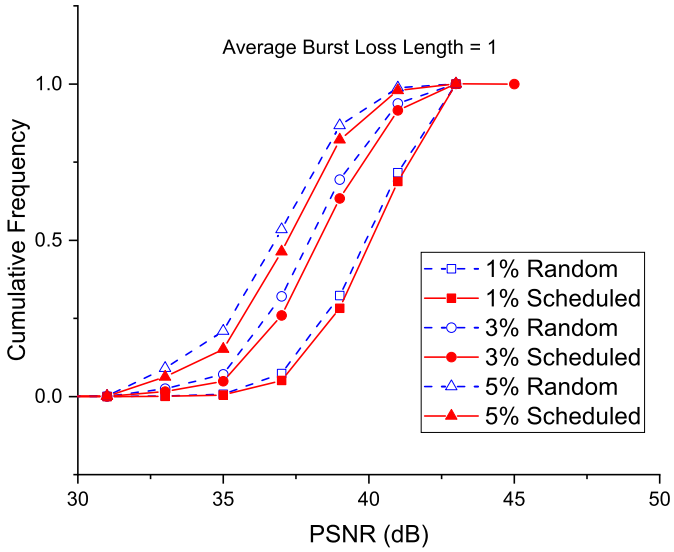


Figure 4.25: CDF of the recovered frames' PSNR with/without the proposed scheduling algorithm when the average burst loss length is 1. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

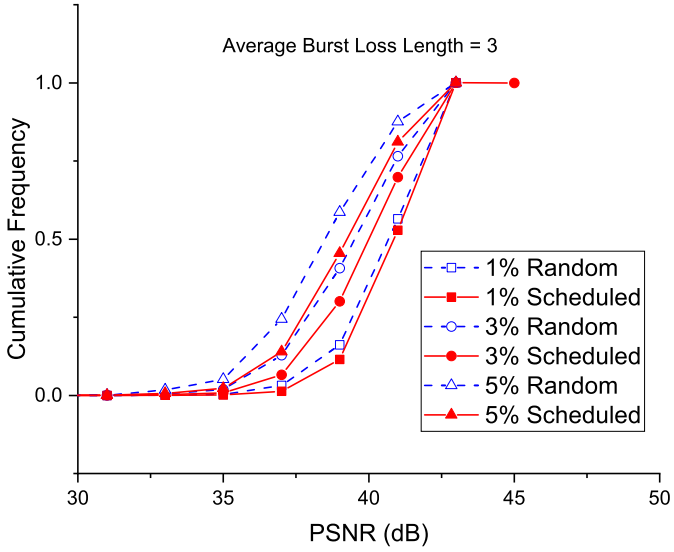


Figure 4.26: CDF of the recovered frames' PSNR with/without the proposed scheduling algorithm when average burst loss length is 3. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

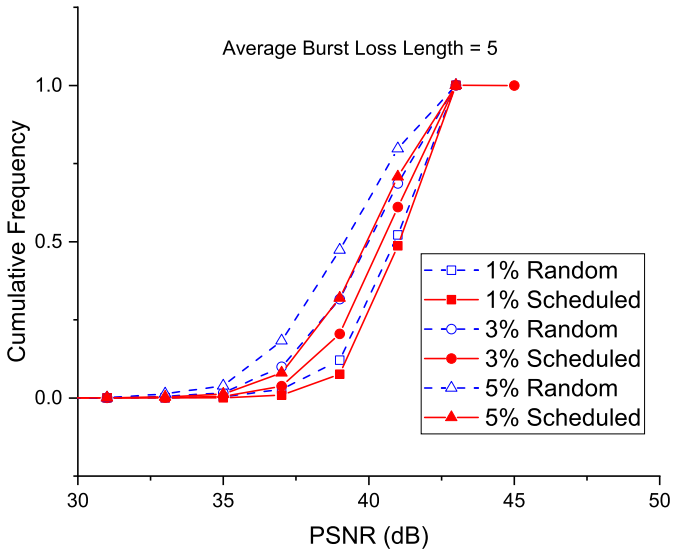


Figure 4.27: CDF of the recovered frames' PSNR with/without the proposed scheduling algorithm when average burst loss length is 5. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

burst loss lengths, the proposed scheduling algorithm can effectively improve the overall level of frame quality at the receiver. After many statistical experiments, it was determined that the proportion of samples whose reconstruction results are improved by using the scheduling algorithm is between 3% and 15.5%. At the same time, it should be noted that the average burst loss length has a greater impact on the transmission of LF MDC. As the average burst loss length increases, the gain brought by the proposed scheduling algorithm becomes more significant. When the average burst loss length is 1 and the packet loss rate is 0.05, the proportion of results exceeding 35 dB after using the proposed scheduling algorithm increases by 8%. However, when the average burst loss length is 5 and the packet loss rate remains the same, there is a 15.5% increase in this ratio. Figure 4.26 and Figure 4.27 show that the output result of the scheduling algorithm in the case of a 5% packet loss rate is close to or even better than the random transmission result in the case of a 3% packet loss rate. The above experimental results prove that the proposed scheduling algorithm can effectively help the LF MDC scheme to improve its resistance to network packet loss.

4.5.2.4 Adaptability to Sudden Network Fluctuations and Network Instability

Here, we show that the proposed solution can make timely adjustments and adaptations when sudden network fluctuations occur. The performance impact of the proposed delivery scheme when random packet loss and a sharp drop in network bandwidth occurs is simulated. As shown in Figure 4.28, transmission tests on the sequence *boxer* are conducted. The situation of packet loss or transmission timeout by randomly discarding some descriptions at the receiver is presented. At the same time, the sharp drop in bandwidth when the network is congested by controlling the upper limit of the receiver's throughput is also simulated. The test randomly discards part of the received description between 7.5 s and 12 s after the start of transmission. At 12 s, the upper limit of the throughput starts to decrease, as shown by the blue line in the figure. The red line indicates the real-time output of the LF video's quality during transmission. We can see that after starting to simulate packet loss, the PSNR output of the proposed scheme starts to greatly fluctuate. Despite this, the proposed solution can still recover all the view stream data at every moment and provide the highest quality output possible. When the bandwidth is greatly limited, although the proposed scheme will sacrifice the quality of some output results, it will still keep working and recover the data of all viewpoint streams. Once the bandwidth situation improves, the PSNR value of the output result can quickly return to the peak state.

The throughput in Figure 4.28 appears to have a significant fluctuation before the drop, which may provide a possibility for throughput prediction. However, in the proposed scheme, the design of prediction and active service is not adopted. This is mainly because light field video is more commonly used for 3D perception and modeling of the scene, rather than being directly watched by the user. In the traditional video transmission process, an incorrect bandwidth prediction may simple cause a frozen rebuffer during the user's viewing process; however, this may lead to the complete failure of the subsequent application of LF. The reason to adopt the MDC idea is to control this risk as much as possible in the encoding and transmission process and not pass it on to the next step. Finally, it will be proven that the proposed solution can not only adapt to sudden network fluctuations but can also maintain very good adaptability in a continuously unstable network environment. Figure 4.29 to Figure 4.32 show the performance comparison between the proposed scheme and SHEVC and MDCNN under different packet loss rates.

The four figures correspond to the four sets of experiments performed on the *cats* sequence. The bandwidth trace record used for each test and the parameter settings of all schemes are exactly the same, and only the network packet loss rate during the simulation process is changed. To simulate a more practical network environment, the average burst

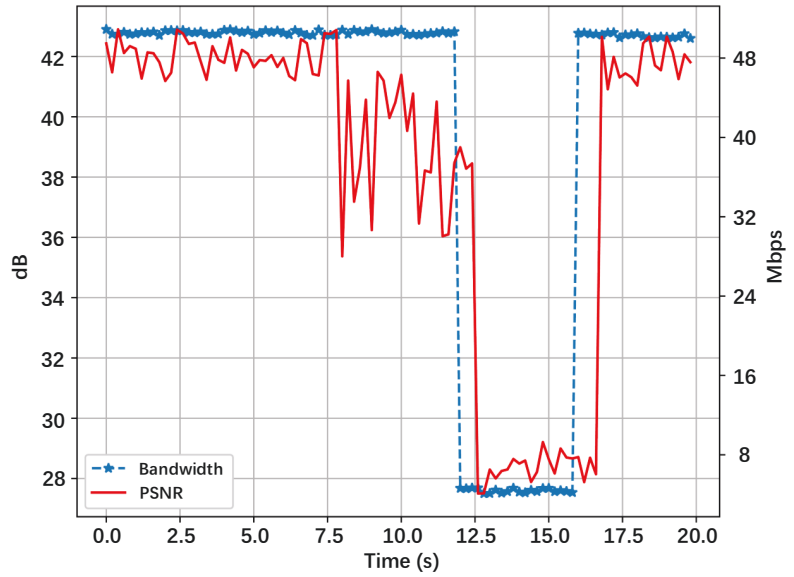


Figure 4.28: Bandwidth and PSNR on *boxer* using the proposed scheme. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

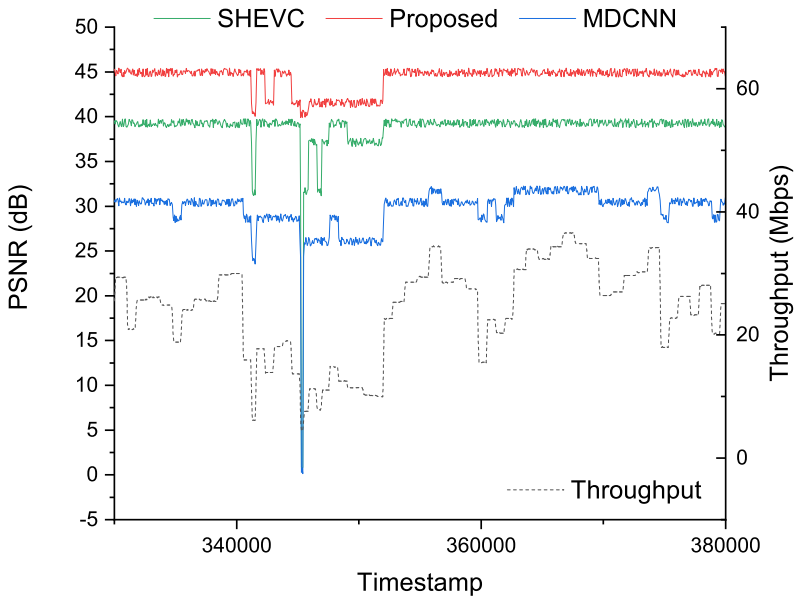


Figure 4.29: Simulation results of the proposed, SHEVC and MDCNN schemes under a 0% packet loss rate. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

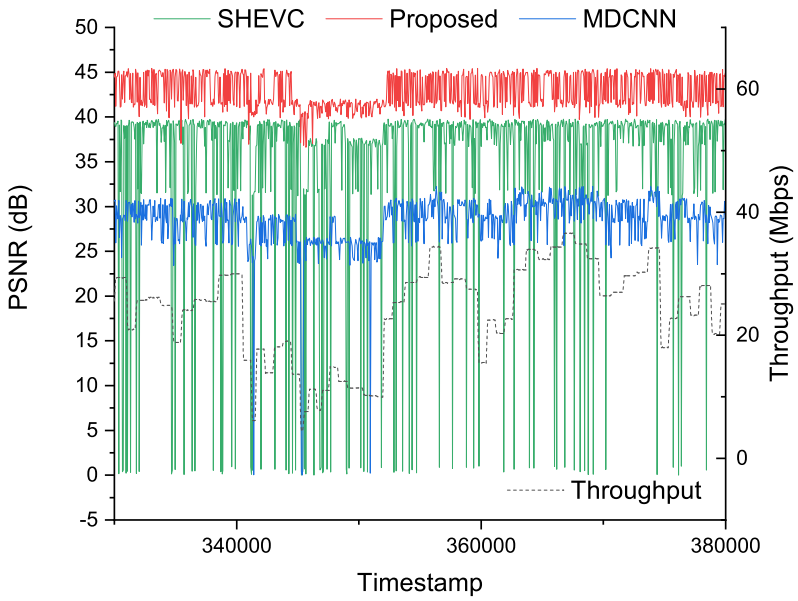


Figure 4.30: Simulation results of the proposed, SHEVC and MDCNN schemes under a 1% packet loss rate. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

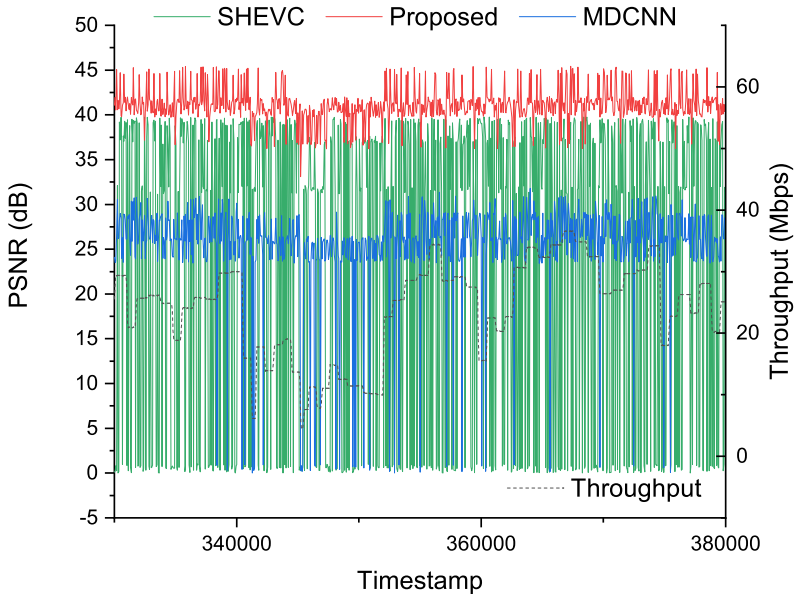


Figure 4.31: Simulation results of the proposed, SHEVC and MDCNN schemes under a 3% packet loss rate. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

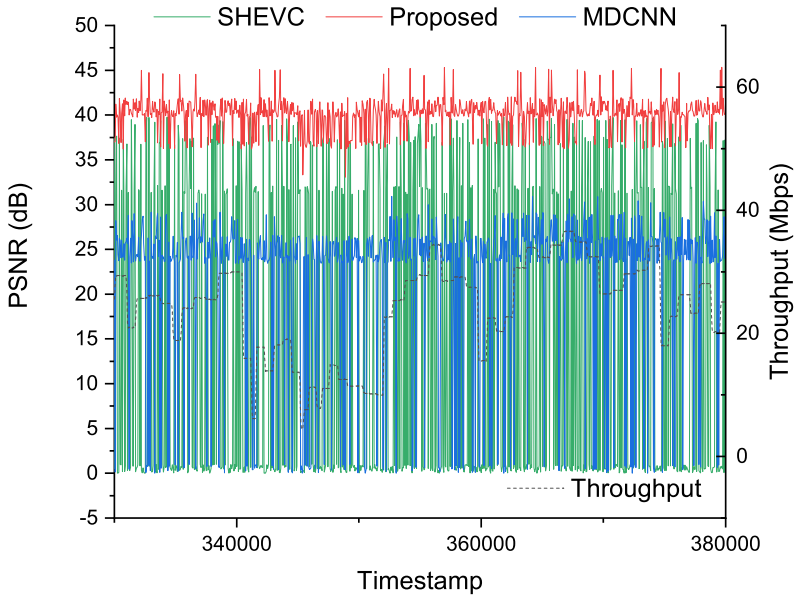


Figure 4.32: Simulation results of the proposed, SHEVC and MDCNN schemes under a 5% packet loss rate. (Reprinted from [J3], ©2021 IEEE, reused with permission.)

loss length in all tests is set to 1. The black dotted line represents the network bandwidth, and its value is mapped on the right vertical axis. The red, green, and blue solid lines correspond to the restored frame's real-time PSNR of the proposed, SHEVC and MDCNN schemes at the receiver, respectively, and their values are mapped to the left vertical axis.

Figure 4.29 shows that without considering the packet loss rate, both SHEVC and MDC-based schemes can adapt to bandwidth fluctuations very well. The gap between the three schemes due to compression performance is also consistent with the results from Figure 4.18 to Figure 4.24. However, with the introduction and continuous improvement of the packet loss rate, it should be noted that very different outcomes are observed for SHEVC and MDC-based schemes. Although SHEVC considers redundant coding of the base layer data, it is clear that the excessively high packet loss rate has exceeded its tolerance limitation. Therefore, SHEVC fails to decode many frames. However, in frames that can be normally decoded by SHEVC, their PSNR is still maintained at a high level. In contrast, the proposed scheme is completely different. As the packet loss rate increases, the curves of the proposed scheme slowly decrease overall, but it is ensured that the receiver recovers the frame most of the time, even though the quality is slightly reduced. This is mainly due to the good robustness of the proposed GNN-based compression algorithm, which can avoid the impact of the loss of a small number of viewpoints on the LF video

matrix. In extreme cases, the proposed solution also attempts to restore each frame and avoid the situation of complete decoding failure.

Finally, Figure 4.32 shows that although all tested methods are based on MDC, the average PSNR of the proposed scheme is better than that of the state-of-the-art MDCNN scheme. When the bandwidth very severely drops and the packet loss rate is as high as 5%, MDCNN is unable to continuously recover the image; however, the proposed scheme can still maintain an average output result of approximately 38 dB. This is mainly due to the proposed best-effort adaptive scheduling algorithm, which can improve the receiver's ability to recover high-quality results by optimizing the data transmission sequence. Table 4.3 shows the detailed information of the reception of descriptions in a short period of time and the combination of the highest quality description that can be obtained. These data are taken from the experimental results corresponding to Figure 4.32, which show the detailed workflow of the proposed LF MDC scheme. From the experimental results, it can be seen that only when the throughput is lower than the basic description transmission requirement (0.17 Mbps), or when no basic description can be correctly received due to packet loss, the proposed scheme may completely fail to decode. For example, in the second row of Table 4.3, when the throughput of 0.47 Mbps exceeds the minimum requirement, the sender sends out two basic descriptions. However, due to the packet loss rate, these descriptions were not successfully received by the receiver, so decoding still failed. Except for these extreme situations, the LF-MDC scheme can guarantee the reliability of light field video transmission most of the time.

4.6 Chapter Summary

In this chapter, an MDC scheme is proposed for LF video delivery using GNN-based compression. Unlike most existing static LF image compression and coding work, more attention is given to the design of adaptive and efficient transmission schemes for dynamic LF video sequences in this work. The LF video is downsampled to different levels of description according to different downsampling ratios, and the GNN-based compression method is used to ensure that all descriptions can be independently decoded and recovered. Although MDC's original idea is to treat every description equally in transmission, it is not adopted in this chapter because it is difficult to achieve in real life. The proposed LF MDC scheme implements a pragmatic branch that deviates from the original MDC concept. The basic descriptions cannot be combined at will, some basic descriptions are more likely to be used to compose higher-level descriptions, which makes them more important when reconstructing the high-quality LF matrix. This design is based on the practical consideration

Table 4.3: Reception and combination results of descriptions in real time.

Throughput(Mbps)	Received Descriptions	Best Combination	Highest Level Output
0.11	Null	Null	Decoding Failed
0.47	Null	Null	Decoding Failed
3.77	$[D0, D2]$	$D0$	1/16 Basic Description(16-1)
3.98	$[D0, D1, D2]$	$D0 \sim D1$	1/8 Description(8-1)
9.16	$[D0, D1, D2, D4, D5, D6, D7]$	$D4 \sim D7$	1/4 Description(4-2)
13.92	$[D0, D1, D2, D3, D4, D5, D6, D7, D8, D9, D10, D13]$	$D0 \sim D7$	1/2 Description(2-1)
21.60	$[D0, D1, D5, D6, D7, D8, D9, D10, D11, D13, D14]$	$D8 \sim D11$	1/4 Description(4-3)
21.84	$[D0, D5, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15]$	$D8 \sim D15$	1/2 Description(2-2)
20.88	$[D0, D1, D2, D3, D4, D5, D6, D7, D8, D9, D10, D11, D12, D13, D14, D15]$	$D0 \sim D15$	Full Descriptions
21.96	$[D0, D1, D2, D5, D6, D7, D8, D9, D10, D12, D13, D15]$	$D0 \sim D1$	1/8 Description(8-1)
22.20	$[D0, D1, D3, D4, D5, D6, D7, D9, D10, D11, D13, D14, D15]$	$D4 \sim D7$	1/4 Description(4-2)

of reducing the workload required to retrain the GNN model. On the basis of LF MDC, a scheduling algorithm is designed according to the proposed separation method, and a search is performed for the optimal sending sequence for all basic descriptions. Experimental results prove that the proposed scheme achieves the expected design goal and can effectively recover from packet loss/error caused by the best-effort network.

Chapter 5

Optical-Flow-based AI-Assisted Depth Estimation for Narrow- and Wide-Baseline LF

5.1 Architectural Overview

Figure 5.1 shows the overall framework of the proposed scheme, whose input contains the original SAI matrix and the optical flow matrix predicted from it. It should be noted that this work does not cover or propose an optical flow estimation algorithm; any suitable state-of-the-art optical flow prediction algorithm can be used. Since the main role of the optical flow input is to guide the generation of uniform patches, its prediction accuracy has minimal effect on the precision of the final output of the proposed method. The optical flow prediction result only needs to meet the requirement of basic coarse-grained accuracy. According to the geometric position relationship of the lens matrix, the proposed algorithm can eliminate most of the influence of optical-flow estimation distortion when calculating the patch offset. The algorithm first divides the patch seeds into fixed-size patch seeds in the spatial domain according to the central SAI. For each patch seed, shown as a red solid box in the figure, the proposed algorithm calculates the optimal patch offset according to the optical flow of the corresponding region in all SAIs. The patch seed and patch offset can then be used to generate a uniform patch matrix with a smaller disparity range, which will be more conducive to high-precision depth estimation at the pixel level.

The depth estimation network of the proposed method is mainly composed of two blocks. The EPI block processes the uniform patch matrix into four sets of patch arrays,

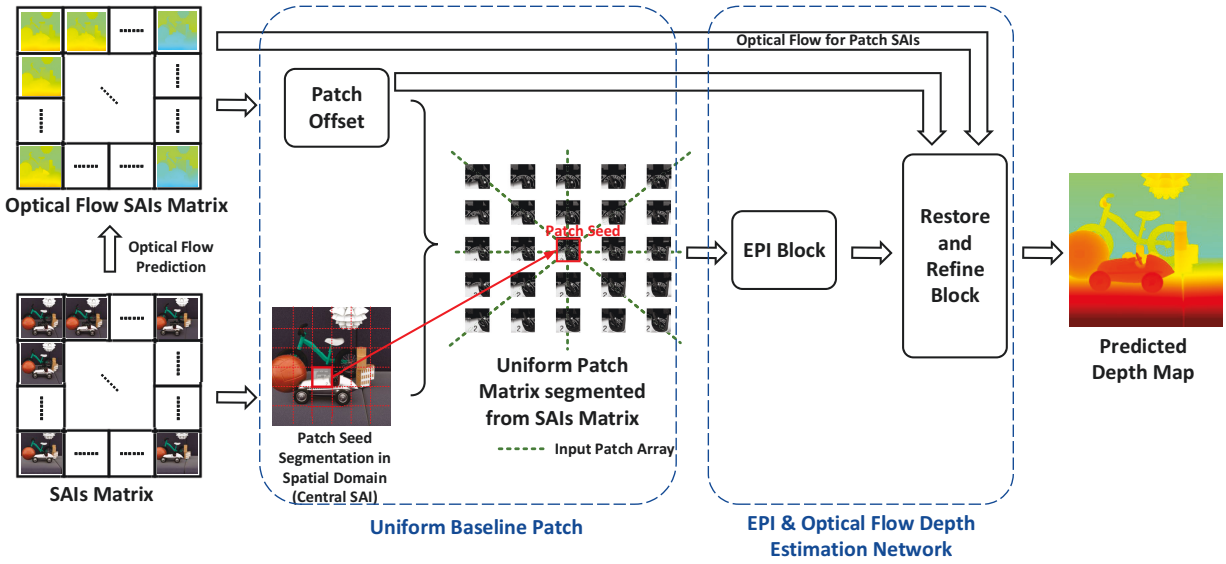


Figure 5.1: Optical-flow-based depth estimation for uniform-baselines LF. (Reprinted from [J1], submitted to IEEE for peer review.)

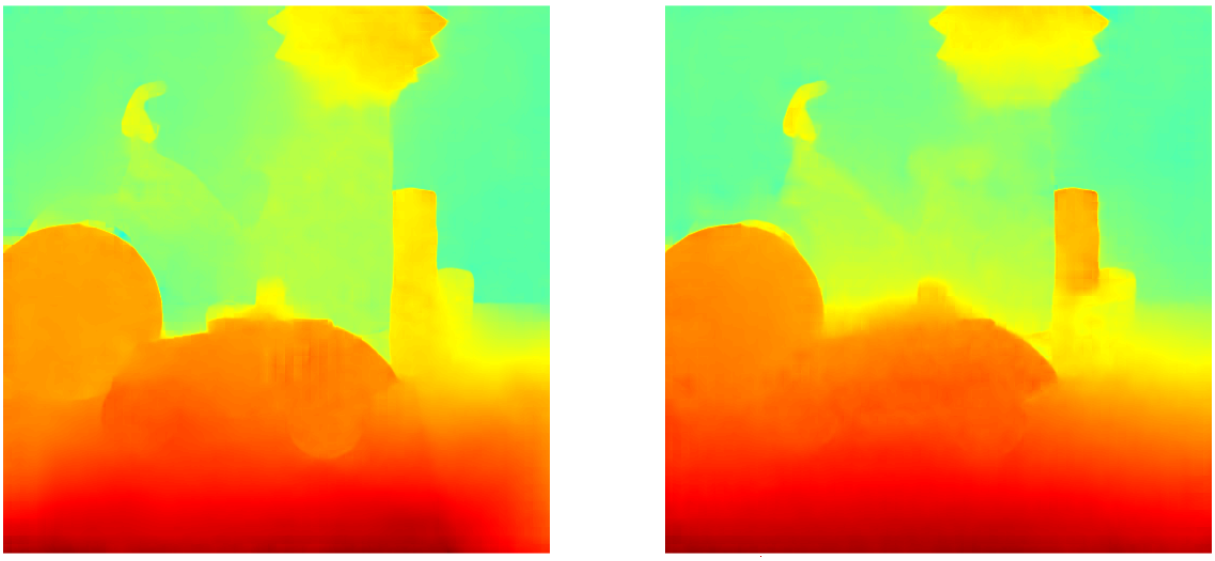
which are the row, column, and two diagonal lines where the central patch is located, indicated by intersecting green dashed lines in Figure 5.1. These four sets of patch arrays correspond to the horizontal, vertical, and positive and negative 45-degree EPI images of the LF. The EPI block can perform pixel-level disparity estimation on the uniform patch matrix, but the estimation result is not the disparity of the original SAI matrix. Therefore, the algorithm needs to restore the disparity output of the EPI block according to the patch offset and the input optical flow, which is a function of the restore and refine block. Finally, the predicted depth map of the central SAI is obtained by recombining the estimated output of all patch seeds. Next, the design of each part of the proposed scheme will be introduced in detail.

5.2 Uniform Baseline Patch Based on Optical Flow

Optical flow is often used to detect the motion of objects in video frames, for example, when determining the motion vector of macroblocks in video coding. Optical flow estimation can also greatly improve applications such as target detection and tracking. Usually, optical flow estimation uses two adjacent video frames as input because the background between adjacent frames minimally changes while the motion of foreground objects is more

significant. The motion difference between the foreground and background objects in a monocular video frame is very similar to the disparity distribution in LF. As mentioned in Figure 2.5 and Equation (2.2), the disparity D_v of the light source O in the SAI matrix is proportional to its depth value D, and the greater the depth is, the greater the disparity, and vice versa.

It may seem intuitive to directly use the optical flow prediction algorithm to estimate the disparity between SAIs, but its effect is not ideal, as shown in Figure 5.2.



(a) Horizontal direction (b) Vertical direction

Figure 5.2: Optical flow prediction result between the central SAI and its upper-left SAI. (Reprinted from [J1], submitted to IEEE for peer review.)

In this figure, the optical flow results in both the horizontal and vertical directions are predicted using the RAFT algorithm proposed by Teed et al. [73]. We can see from Figure 5.2 that compared to the final output depth map given in Figure 5.1, the optical flow results make it difficult to meet the requirements for the detection of distant objects and the clarity of object outlines. However, the basic distribution in the spatial domain of the optical flow value and the disparity value is relevant, so the optical flow between SAIs can be used as a priori information to guide accurate disparity and depth estimation. The input optical flow SAI matrix is obtained by making optical flow predictions for all SAIs and the central SAI one by one. They will be used as the reference to calculate the optimal offset for all patch seeds.

Since the lenses of the LF acquisition matrix are equally spaced in the horizontal and vertical directions, the disparity distribution of each SAI is related to its position in the SAI matrix, as shown in Figure 5.3. Let $SAI_{(s,t)}$ represent the SAI patch captured by the

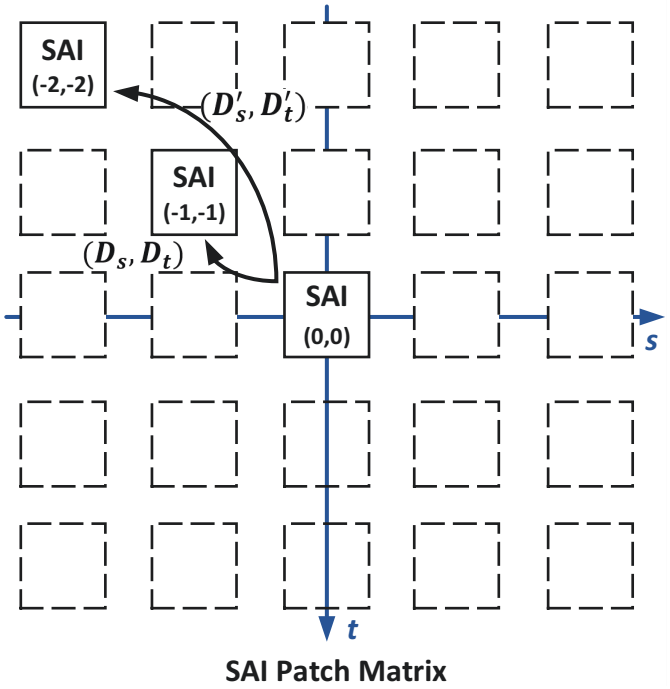


Figure 5.3: Relationship between SAI patch disparity and its angular domain position. (Reprinted from [J1], submitted to IEEE for peer review.)

lens at the position of the angular domain coordinates (s,t) so that the center SAI patch is defined as the origin $SAI_{(0,0)}$. (D_s, D_t) and $(D_s^{'}, D_t^{'})$ are the disparities between $SAI_{(-1,-1)}$ and $SAI_{(-2,-2)}$ and the center SAI patch, respectively. According to Equation (2.2), we have:

$$\begin{aligned} D_t &= \alpha \times (t(SAI_{(-1,-1)}) - t(SAI_{(0,0)})) \times B \\ D_t^{'} &= \alpha \times (t(SAI_{(-2,-2)}) - t(SAI_{(0,0)})) \times B \end{aligned} \tag{5.1}$$

where $\alpha = \frac{D}{F}$, and $t(SAI)$ is the vertical coordinate of the SAI in the angular domain. Therefore, D_t should be twice $D_t^{'}$, and their proportional relationship depends on the relative positions of their corresponding SAI patches in the angular domain. In the same way, the disparity between SAI patches in the horizontal direction can also be obtained. The relationship between the disparity of the SAIs and their relative positions in the angular domain is the basis for the LF to use the EPI method for depth estimation, so it

is necessary to retain the EPI characteristic as much as possible during the processing of the uniform patch matrix.

To reduce the effect of the baseline B so that all patch matrices can be processed to the same disparity level for prediction, patch offset S is introduced to remove the main components of the original disparity during the generation of the patch matrix. Equation (5.2) gives the definition of the reduced SAI patch disparity d_t.

$$\begin{aligned} d_t &= D_t - \Delta t \times S \\ &= \alpha \times \Delta t \times (B - \frac{S}{\alpha}) \end{aligned} \tag{5.2}$$

where d_t has a smaller distribution range, and the aim is to retain the EPI characteristic of the original D_t. The selection of an ideal patch offset S should satisfy two requirements:

1. S needs to be constant so that uniform translation operations can be performed on all the pixels in the patch.
2. The subtraction term for baseline $\frac{S}{\alpha}$ should be unchanged on all pixels in the patch so that the EPI characteristic of D_t can be completely preserved.

However, this is problematic because α is not constant within a patch; it is affected by the depth D of the light source, as shown in Equation (5.1). To explore the optimal constant patch offset S, the definition of d_t from pixels to patches must first be generalized, as shown in Equation (5.3).

$$\begin{aligned} d_t(u,v) &= \frac{D(u,v)}{F} \times \Delta t \times (B - \frac{S \times F}{D(u,v)}) \\ &= \frac{D(u,v)}{F} \times \Delta t \times B - \Delta t \times S \end{aligned} \tag{5.3}$$

where (u, v) represents the spatial coordinates of the pixels in the patch, and F is invariant in both the LF angular domain and the spatial domain because the LF acquisition matrix generally uses fixed focal length lenses. According to Equation 4, an unbiased estimate of the depth value can be obtained:

$$D(u,v) = \frac{(d_t(u,v) + \Delta t \times S) \times F}{\Delta t \times B} \tag{5.4}$$

To minimize the range of $d_t(u, v)$, the problem can be transformed into solving its

minimal L1 normal form as follows:

$$\arg\min_{S} \quad \|d_t(S)\|_1 \iff \arg\min_{S} \quad |\frac{D(u,v) \times B}{F} - S| \tag{5.5}$$

Equation (5.6) gives the optimal S:

$$S = \frac{B \times \bar{D}}{F} \tag{5.6}$$

where $\bar{D}$ is the mean of $D(u,v)$ within the patch. According to Equation 1, the optimal S at this time is the mean value of the disparity in the patch. Because optical flow can be regarded as a rough estimate of the disparity between SAIs, the patch mean of the optical flow $\hat{S}$ is used as the biased estimator of the optimal S. Equation (5.4) proves that even if $\hat{S}$ is not exact, it can still reliably deduce the depth value D from d_t. However, it will cause d_t to not be reduced to the minimum range, thus increasing the difficulty of subsequent EPI estimation.

The above discussion is based on one group of SAIs for analysis, and more groups of SAIs can be considered together to further improve the reliability of patch offset value selection. Let us define the pixel resolution of the patch as $N_S \times N_S$ (spatial resolution) and the SAI number of the SAI matrix as $N_A \times N_A$ (angular resolution). Equation (5.7) shows the definition of patch offset S_{global} by weighted averaging in the angular domain.

$$\begin{aligned} S_{global} &= \frac{\sum_{s\in[0,N_A-1],t\in[0,N_A-1]} G_u(s,t) + G_v(s,t)}{\sum_{s\in[0,N_A-1],t\in[0,N_A-1]} |s - \frac{N_A-1}{2}| + |t - \frac{N_A-1}{2}|} \\ G_u(s,t) &= (s - \frac{N_A-1}{2}) \times \frac{\sum_{u\in[0,N_s-1],v\in[0,N_s-1]} O_u(s,t)}{N_S^2} \\ G_v(s,t) &= (s - \frac{N_A-1}{2}) \times \frac{\sum_{u\in[0,N_s-1],v\in[0,N_s-1]} O_v(s,t)}{N_S^2} \end{aligned} \tag{5.7}$$

where O_u and O_v are the optical flow prediction results in the horizontal and vertical directions, respectively. It should be noted that the weighted average in the angular domain cannot eliminate the error of the input optical flow matrix, but it can reduce the possible impact of a single SAI optical flow prediction failure on the patch offset and improve the robustness of the proposed scheme to the optical flow input quality.

When the appropriate patch offset is determined, the generation process of the uniform patch matrix is shown in Figure 5.4. The left image is the central SAI, and the red box

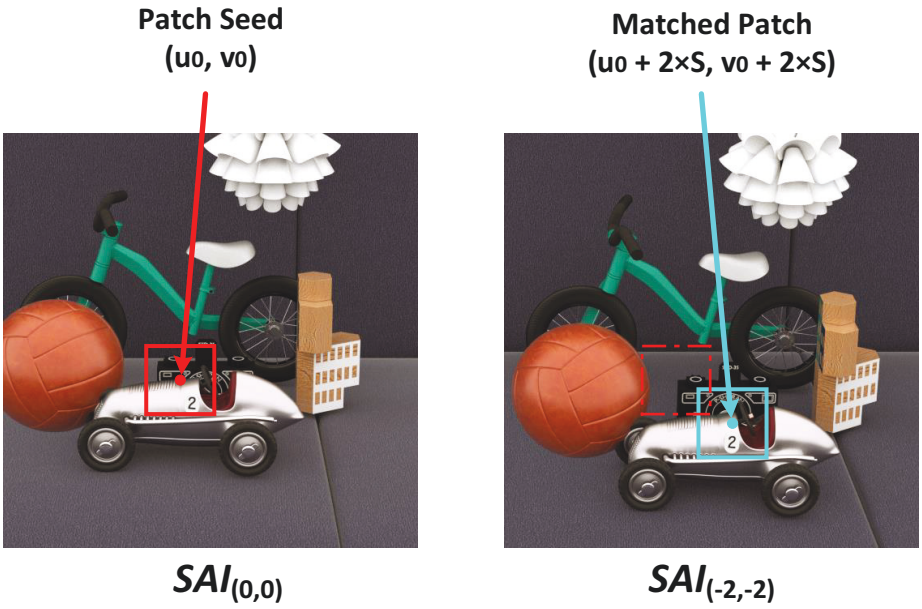


Figure 5.4: Uniform patch matrix generation based on patch offset S. (Reprinted from [J1], submitted to IEEE for peer review.)

represents the patch seed centered at (u_0, v_0). The optimal patch offset of the patch seed can be calculated by the input optical flow matrix S. The figure on the right shows how to locate the matching patch in $SAI_{(-2,-2)}$. The center coordinates of the matching patch should be shifted by $2 \times S$ pixels both horizontally and vertically, as shown with the turquoise box in the figure. The above process needs to traverse all noncentral SAIs in the SAIs matrix once, and the generated uniform patch matrix for the wide-baseline version of *Toy* is shown in Figure 5.5. Comparing Figure 5.5 and Figure 2.9(b), we can see that the proportion of overlapping areas between SAI patches has significantly increased, which means that the disparity of the patch matrix has been effectively reduced. Although the disparity of the wide-baseline LF is significantly reduced in the uniform patch matrix, it is still larger than the disparity of many patch matrices of the narrow-baseline LF (less than 1 pixel). Therefore, when generating the uniform patch matrix for the narrow-baseline LF, the patch offset will be forced to 1 for patch matrices with a disparity of less than 1. The advantage of setting the mandatory minimum value of patch offset is that it can reduce the difference in the uniform patch matrix samples between the wide-baseline and the narrow-baseline LF, which helps to solve the overfitting problem of the subsequent depth estimation network.

Figure 5.5: Uniform patch matrix for the wide-baseline version of *Toy*. (Reprinted from [J1], submitted to IEEE for peer review.)

5.3 Depth Estimation Network Based on EPI and Optical Flow

With the help of the uniform patch matrix, the proposed scheme can mix wide- and narrow-baseline datasets for estimation network training. EPINet [52] is a classic CNN-based prediction network in the field of LF depth estimation, and its reliability on the HCI dataset [74] has been verified. It has the advantages of stable performance and fast convergence, so in the EPI block, this network design is followed to predict the disparity d_t of the uniform patch matrix. The detailed structure of the EPI block is shown in Figure 5.6. The uniform patch matrix is decomposed into 4 streams used as input to the EPI block, and each stream represents a different angle of the LF EPI array (0, 90, 45 and -45 degrees). Streams will first independently pass through 2 convolution modules (Conv1_n and Conv2_n), and ReLU is used as the activation function for each layer.

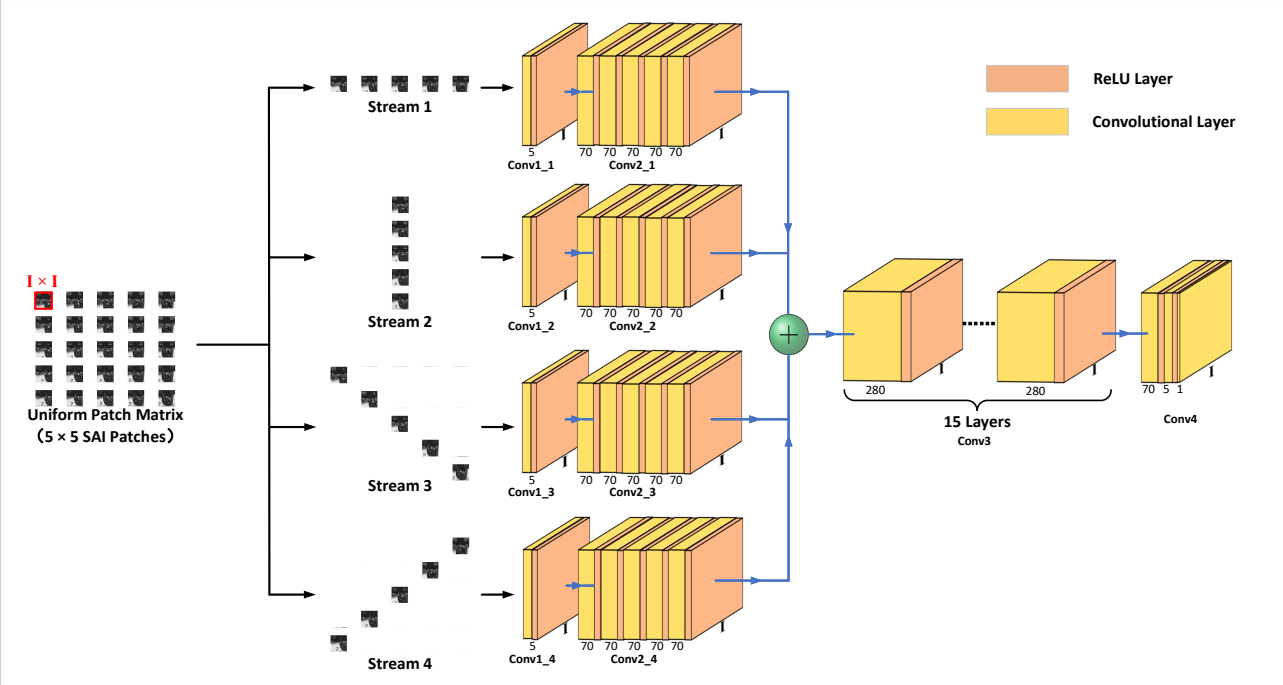


Figure 5.6: EPI block for the proposed depth estimation network. (Reprinted from [J1], submitted to IEEE for peer review.)

Subsequently, the features of the four streams are concatenated and fed into a deeper convolution module. After concatenation, the size of the feature will be 4 times larger. Conv3 is the main part of the EPI block, which contains 15 convolutional layers. Unlike in EPINet, a batch normalization layer is not added between the convolutional layers in the proposed network. This is mainly because in the experiments, it can be found that with the help of the uniform patch matrix, the proposed network did not encounter a serious overfitting problem. Therefore, there is no need to perform batch normalization anymore. Finally, Conv4 reduces the 280-channel features to one channel as the prediction output of d_t. It should be noted that since the disparity has both positive and negative values, the ReLU activation function is not used in the last convolutional layer.

In the restore and refine, the estimated d_t is first restored to the original disparity D_t of the SAI patch using a patch offset S according to Equation (5.2). The detailed structure of the restore and refine block is presented in Figure 5.7. Patch offset S will be added to all pixels of the EPI block output feature, thereby mapping d_t back to the original disparity range. Then, the optical flow results are introduced as a reference for disparity-refined estimation. However, it is not ideal to introduce too much optical flow information, as the

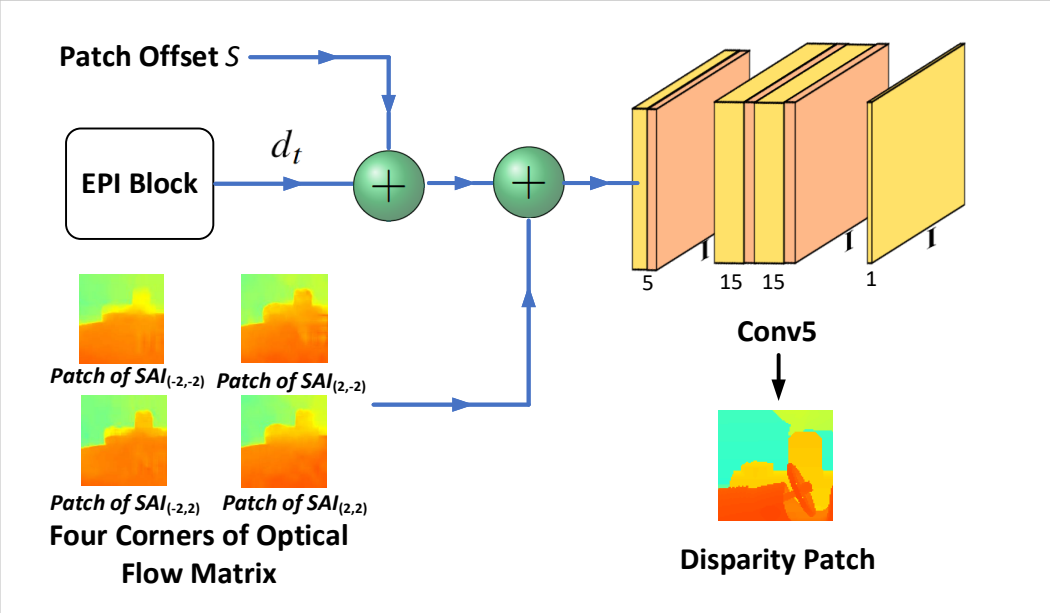


Figure 5.7: Restore and refine block of the proposed depth estimation network. (Reprinted from [J1], submitted to IEEE for peer review.)

prediction results may become degraded by optical flow. Therefore, only the SAIs of the four corners of the optical flow matrix in the restore and refine block, which are $SAI_{(-2,-2)}$, $SAI_{(-2,2)}$, $SAI_{(2,-2)}$ and $SAI_{(2,2)}$, are introduced. There are two reasons for choosing these SAIs: first, they are the farthest from the central SAI in the angular domain and contain the most significant motion information, which can help correct possible incorrectly predicted pixels in d_t. Second, these corner SAIs have valid optical flow values in both the horizontal and vertical directions, and theoretically, their absolute values should be the same. This allows us to easily take into account the optical flow results in both directions by averaging the horizontal and vertical optical flow predictions. The four optical flow patches and the restored d_t are concatenated together to form a 5-channel feature, which is next sent to the Conv5 convolution module. Similar to the EPI block, the final output convolutional layer does not contain the ReLU activation function.

5.4 Data Argumentation

Different from traditional 2D image datasets, LF datasets, especially those with depth maps, are usually small in size, generally containing only dozens of sets of LF images. For example, the training set of the commonly used HCI dataset [74] contains only 16 sets

of LF images and only narrow-baseline LF samples. Such a limited number of training samples obviously makes it difficult for the proposed network to converge well, so data augmentation is needed. Two strategies are used in this work to meet the needs of data augmentation.

The first data augmentation strategy is to generate samples with patch seeds at random positions during training. The patch seed generation method shown in Figure 5.1 uniformly segments the central SAI, but this method can only generate $\frac{Size(SAI)}{Size(Patch)}$ patch samples with each set of LF images. This is not sufficient to fundamentally solve the problem of insufficient training samples. The proposed solution is to change the method of presegmenting all patch training samples before training to generate each patch sample in real time based on the input LF and optical flow matrix during training. Each time a sample is generated, a square patch seed is constructed and centered on a random pixel of the central SAI. When part of the patch seed area exceeds the SAI boundary, the nearest neighbor padding method is used to expand the SAI. This method is also applicable to solve the problem if the matched patch of a certain SAI exceeds the boundary in the process of generating the uniform patch matrix. By randomly selecting patch seeds in real time, a set of LF images can theoretically generate $Size(SAI)$ patch samples. The number of training samples has been greatly increased.

The second strategy is to transpose the LF image in four-dimensional space. The two 2D transposition operations are performed on the original LF image $L(u, v, s, t)$ in the spatial and angular domains to obtain the transposed LF image $L^T(v, u, t, s)$. If only one transpose operation is performed in the spatial or angular domain, the disparity direction between SAIs will be reversed. The connection order between the sample streams and the network input needs to be changed to maintain the direction consistency of the network-predicted result. This obviously has a high implementation cost, so only the method of two transpose operations, which does not require any changes to the network structure, is considered.

After adopting the above two data augmentation strategies, the number of training samples can already meet the actual needs of the proposed network.

5.5 Experiments

5.5.1 Experimental Setup

In this section, the performance of the proposed algorithm is evaluated using the HCI [74] and Inria [16] datasets. The HCI dataset is a popular narrow-baseline LF dataset used in

many works related to LF depth estimation. Its training set contains 16 LF images, and the test set contains 4 LF images. The disparity range of the HCI dataset does not exceed 4 pixels. The Inria dataset is a synthetic LF dataset rendered by the Blender engine and consists of a sparse LF part and a dense LF part. The dense LF part is considered to be narrow-baseline LF data, and its disparity range is similar to that of the HCI dataset, while the sparse LF part corresponds to wide-baseline LF data with a maximum disparity between 10 and 20 pixels. In this work, the sparse and dense LF images of the Inria dataset are mixed for training and testing. Inria has 55 LF images for training, 12 for testing, and 3 for validation. The LF images of both datasets are uniformly processed in the form of 5×5 angular resolution and 10×10 spatial resolution. The main experiments in this chapter are performed on the Inria dataset, and the HCI dataset is only used to verify generalization.

For the proposed depth estimation network, the mean absolute error (MAE) is used as the loss function. The RMSProp [75] optimizer is used, and the batch size is set to 1. The learning rate is set to 1e-5 and then is decreases to 1e-6. The proposed method is implemented using Python 3.7 and PyTorch 1.6.0, and it takes an average of 2-3 days to complete a training session on an NVIDIA GTX 1080 TI.

5.5.2 Quantitative Evaluation of Prediction Accuracy

The mean square error (MSE) and bad pixel (BP) ratio on the test set of the Inria dataset are used to evaluate the accuracy of the algorithm. Since the disparity range of wide- and narrow-baseline LF is usually not on the same order of magnitude, 5% of the ground truth's maximum is taken as the threshold of the bad pixel ratio. Figure 5.8 and Figure 5.9 show the estimation results and quantitative quality results using the four LF depth estimation schemes, including the proposed method, on the test set. At the top of the figures, the disparity range, ground truth, and center SAI for each group of LFs are provided. Three state-of-the-art LF disparity estimation schemes are selected for comparison. Shin's scheme [52] is EPINet, which is a classic narrow-baseline LF depth estimation network. The schemes of Chuchvara [17] and Leistner [14] are both designed for wide-baseline LF. The red and green boxes represent the optimal and suboptimal results on MSE and BP, respectively.

Figure 5.8 shows the estimation results of the wide-baseline LF in the test set: *Furniture*, *Lion*, *Sculpture* and *Flying_furniture*. It should be noted that in the actual training and testing process, the wide- and narrow-baseline LF are used together, and they are only shown separately here for the convenience of observation and comparison of experimental

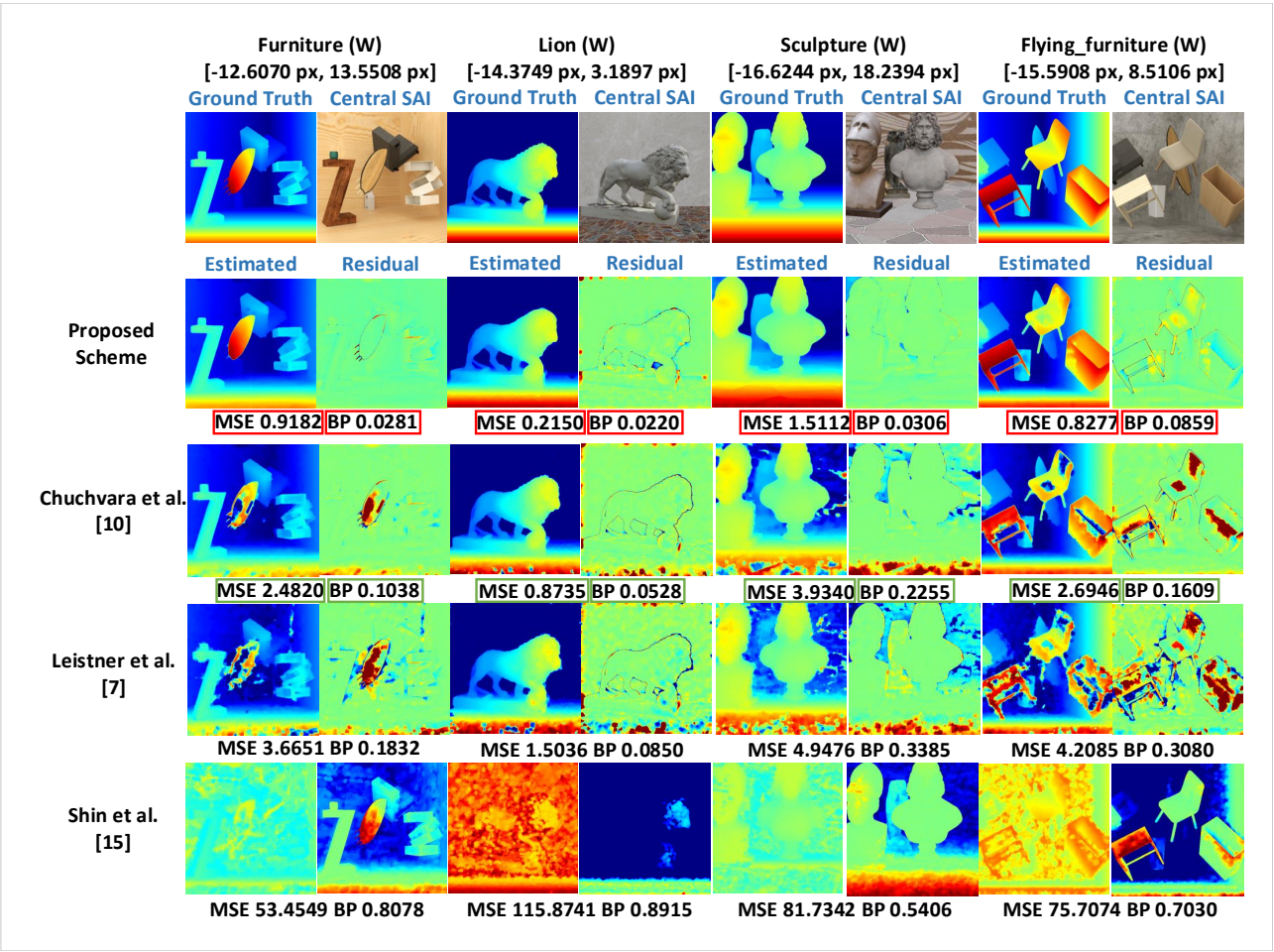


Figure 5.8: Estimation results on the wide-baseline LF: *Furniture*, *Lion*, *Sculpture* and *Flying_furniture*. (Reprinted from [J1], submitted to IEEE for peer review.)

results. The experimental results show that the proposed scheme can still achieve better results on wide-baseline LF even compared with [14] and [17], which are specifically designed and optimized for wide-baseline LF. The results of both MSE and BP are better than those of the other schemes. The result of scheme [52] shows that the narrow-baseline LF depth prediction method does not work at all when dealing with wide-baseline LF, and it can be seen from the residual results that the main components of disparity are not well predicted.

Figure 5.9 shows the estimation results of the narrow-baseline LF in the test set: *Black & white*, *Blue_room*, *Bottles* and *Electro_devices*. Compared with scheme [52], the performance of the proposed scheme is close to that of the narrow-baseline LF scheme. Although

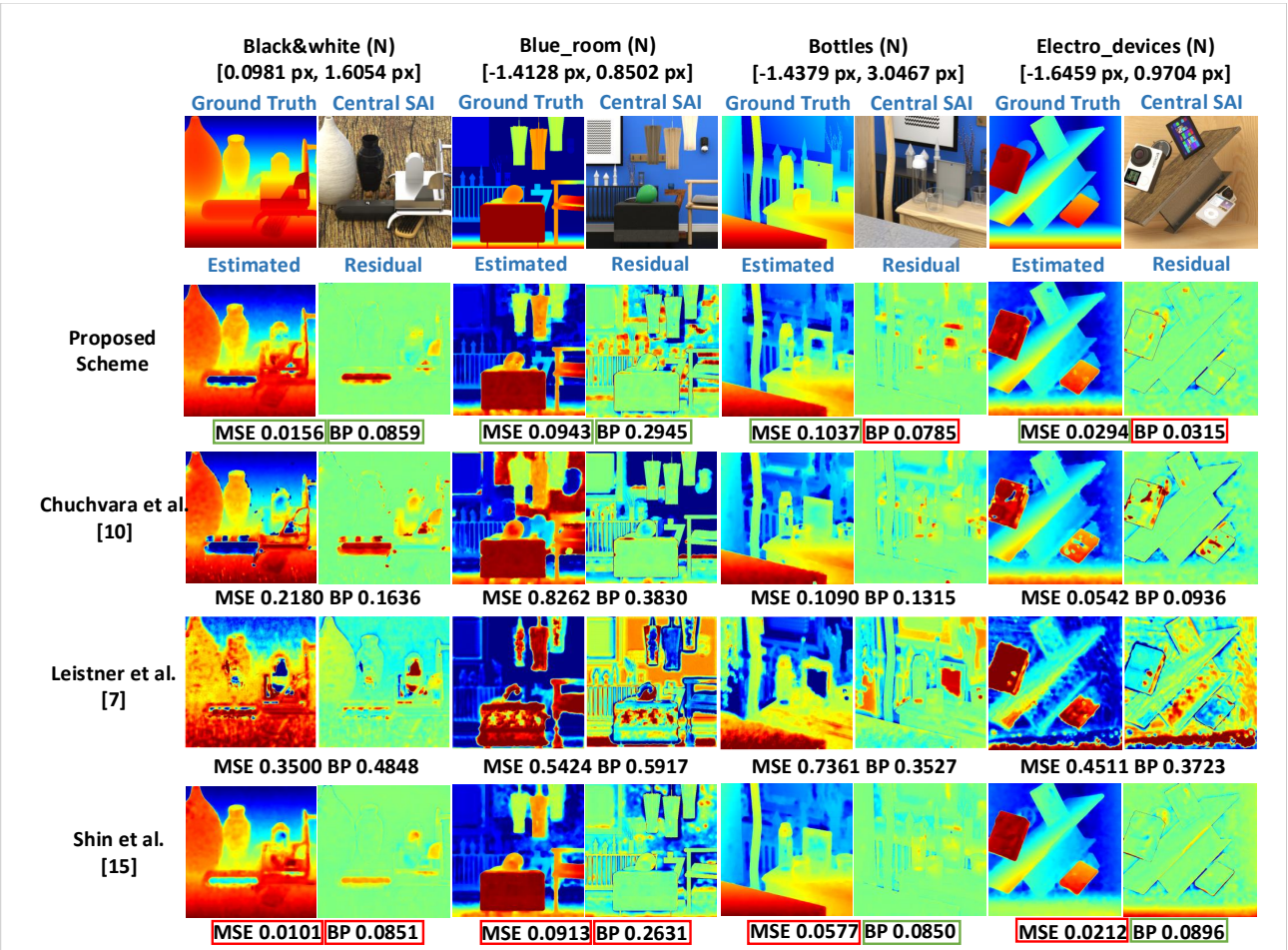


Figure 5.9: Estimation results on the narrow-baseline LF: *Black & white*, *Blue_room*, *Bottles* and *Electro_devices*. (Reprinted from [J1], submitted to IEEE for peer review.)

the proposed scheme is inferior to EPINet on MSE, the gap in the bad pixel ratio is very small or even partially better than that of EPINet. According to the analysis of the residual map, the estimation results of the proposed scheme for good pixel areas are almost indistinguishable from EPINet, but the deviation for bad pixel areas is larger, which is why the BP is similar but the MSE is higher. This result is not unexpected because in the proposed method, generating the uniform patch matrix not only reduces the disparity of the wide-baseline LF but also slightly increases the disparity of the narrow-baseline LF to reduce the sample variability (usually 1 pixel). Therefore, for those regions that are hard to predict, the proposed scheme is more inclined to give a larger predicted value, thereby further increasing the error in these regions. This can be seen as a compromise between

partial samples and the whole sample, where the accuracy of a small number of narrow baseline samples is sacrificed for the improvement of the overall accuracy. The results of [17] and [14] also reflect the existence of this tradeoff; although these two schemes have the ability to predict wide-baseline LF, they ignore the decrease in predictive ability for narrow-baseline LF.

The results in Figure 5.8 and Figure 5.9 show that the proposed scheme achieves the original design intent well; the intent is to design a depth estimation method that can take into account both narrow- and wide-baseline LF and eliminate the influence of baseline differences as much as possible. Here, the idea of multiple models should be discussed. Undoubtedly, if two network models are trained to deal with the narrow- and wide-baseline LF, it should be possible to achieve a better overall performance in terms of statistical indicators. However, this approach lacks practical significance. First, it is impossible for real-world data sources, such as the datasets used in this research, to be aware of their disparity ranges in advance, and there is no one to carefully classify the data into sparse and dense LF before using them. Second, the LF is currently classified as a wide and narrow baseline, but what can be done with boundary samples that may be between narrow and wide baselines? Even if the proposed method additionally trains a new model to process these *intermediate* samples, new boundary samples will be generated, and the cycle repeats. Therefore, it is desirable to first process all samples to the uniform baseline and then perform the depth prediction. It is believed that this is a more promising solution given LF baseline differences.

5.5.3 Generalization and Robustness Analysis

To demonstrate the generalization of the proposed scheme, more experiments on the test set of the HCI dataset are conducted. The experimental results are shown in Figure 5.10. As seen from the figure, the proposed scheme still works well on the HCI dataset. Although the primary goal is not improving the prediction accuracy of the narrow-baseline LF as in other works, to prove that the performance of the proposed scheme is still in an acceptable range for the narrow-baseline LF, it is compared with other schemes on the HCI website ranking. Table 5.1 presents the average MSE results of the proposed scheme and multiple other schemes on the HCI test set. According to the latest HCI rankings, the performance of the proposed scheme on the narrow-baseline HCI dataset is roughly between the 30th and 40th algorithms. At present, a total of 91 algorithms are included on the HCI website, so it can be said that even if the proposed scheme compromises on the wide- and narrow-baseline samples, it can still achieve middle to high estimation accuracy for narrow-baseline LF.

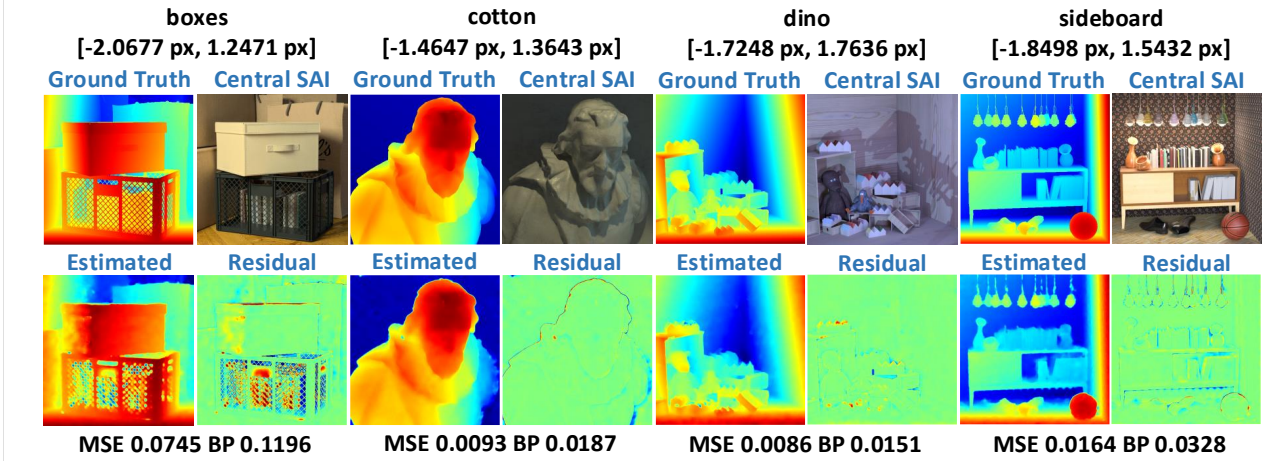


Figure 5.10: Estimation results of the proposed scheme on the HCI test set. (Reprinted from [J1], submitted to IEEE for peer review.)

Table 5.1: Average MSE on the HCI dataset (multiplied by 100)

Algorithm	HCI Rank	AVG MSE
OACC-Net	1	1.698
FEN	10	2.193
OBER-cross+ANP	20	2.584
ESnet	30	3.020
MEPN+OCCNet	40	3.763
Proposed Method		3.383

The proposed scheme is also tested on the author's published real-world dataset 4DLFVD [C1], which currently has the highest number of viewpoints at 1080P among all public LF video datasets. Its acquisition array contains 10×10 cameras with a spatial resolution of 1920×1056. The horizontal and vertical lens baselines are both 35 mm. Since 4DLFVD does not explicitly provide the depth truth, Figure 5.11 only presents part of the central SAI and its corresponding estimated depth map and does not provide the quantitative quality result. It can be seen that the proposed method works satisfactorily on real-worldwide-baseline LF datasets.

Finally, a robustness experiment is conducted on the accuracy of the input LF. First, Gaussian filters are used with different parameters to process the predicted optical flow results, and then, they are used for training and testing. The experimental results are

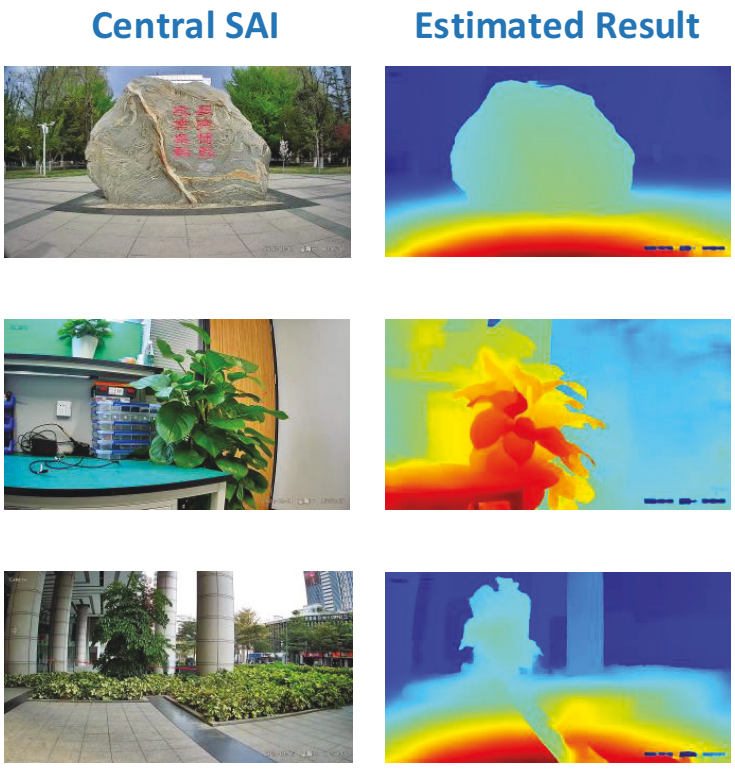


Figure 5.11: Estimation results of the proposed scheme with a real-world dataset. (Reprinted from [J1], submitted to IEEE for peer review.)

shown in Table 5.2. Because the disparity between the narrow- and wide-baseline LF samples of the Inria dataset is very unbalanced, calculating the average MSE of the entire test set does not give good feedback on the overall performance, especially for narrow-baseline samples. Here, the average MSE of the wide- and narrow-baseline samples are separately calculated. Since the same random initial seed is used for each training process, the randomness of training has a negligible effect on the results. It can be seen from Table 5.2 that the Gaussian-filtered optical flow input has a very limited influence on the performance of the proposed scheme. When the Gaussian kernel is increased to 7×7, the average bad pixel rate is only increased by approximately 2%. Furthermore, the mean MSE on the wide-baseline sample is the most variable of all the numerical results, rising from 0.8 to approximately 1.6. However, compared to the results of the other schemes in Figure 5.8, this increase is well within acceptable limits. The experimental results show that the proposed scheme does not depend on the accuracy of the input optical flow, and the coupling between the accuracy of the output-estimated disparity and the input optical

Table 5.2: Input of the optical flow processed with different gaussian filters

Kernel Size	Sigma	AVG BP (Overall)	AVG MSE (Wide)	AVG MSE ×100 (Narrow)
None	None	0.0821	0.8680	6.0750
(3,3)	0.5	0.0843	0.8943	6.1445
(3,3)	1.5	0.0873	0.9722	6.3849
(5,5)	0.5	0.0867	1.1452	6.8362
(5,5)	1.5	0.0882	1.0121	7.4728
(7,7)	0.5	0.0949	1.2462	6.8846
(7,7)	1.5	0.1079	1.6047	7.3105

flow is low.

Through the above experiments, it can be concluded that the proposed scheme has good overall performance in both generalization and robustness.

5.6 Chapter Summary

In this chapter, an optical-flow-based depth estimation scheme is proposed that performs well for both narrow- and wide-baseline LF. Unlike most existing LF depth estimation work, it focuses more on how to provide balanced depth estimation for narrow- and wide-baseline LF. The appropriate patch offset is estimated through the input optical flow, thereby shifting the input LF into a uniform patch matrix with the same disparity range. With the uniform patch matrix, the disparity of the wide-baseline LF patch matrix can be scaled to the same level as the narrow-baseline LF, which makes it possible to use one network to achieve depth estimation for both narrow and wide baselines. On this basis, a multiblock CNN prediction network is designed. It can estimate the intermediate disparity after uniform scaling, restore and refine it with the help of patch offset and optical flow information, and finally output the predicted value of the real disparity. Experimental results show that the proposed scheme achieves the expected balanced performance on the narrow- and wide-baseline LF and has good generalization and robustness.

Chapter 6

Edge-Assisted Virtual Viewpoint Generation for Immersive Light Fields

6.1 LF ViewPoint (VP) Generation

In this chapter, how to apply network edge computing technology to the LF ViewPoint (VP) generation task is mainly discussed. As introduced in Chapter 3, the LF reconstruction technique superresolves the SAI matrix as a whole. However, LF users may not need to generate as many new VPs when they watch LF videos; thus, the VP generation task when viewing LF videos is not the same as that when performing LF reconstruction. This task refers to the use of existing VPs to synthesize a designated single VP requested by users.

When a user's requested viewpoint is already captured by a camera/lens, it is only necessary to deliver that VP to the user. However, when the user requests a VP that has not been directly captured by a camera/lens, a virtual VP needs to be generated by viewpoint interpolation. In Figure 6.1, there is an LF with the VP matrix of $N \times N$. The box with the solid line in the right figure represents the actual directly captured VP, and the box with the dotted line represents the virtual VP to be generated. When the user requests the VP $V_{(0.5,0.5)}$, since none of its adjacent (top, bottom, left or right) VPs are real, this request cannot be directly obtained by interpolation. Therefore, $V_{(0,0.5)}$ and $V_{(1,0.5)}$ must first be obtained by interpolating $V_{(0,0)}$, $V_{(0,1)}$, $V_{(1,0)}$ and $V_{(1,1)}$, respectively, and then generating $V_{(0.5,0.5)}$. Hence, three rounds of interpolation are needed. To reduce the latency and balance the computing load of the user's device, $V_{(0,0.5)}$ or $V_{(1,0.5)}$ can be pregenerated and cached. This can reduce the number of interpolations. However, the

cached viewpoints take a long time to process and store on the user's device, especially when considering that immersive equipment usually requires two VP streams, one for each eye; thus, the overhead mentioned above will be doubled.

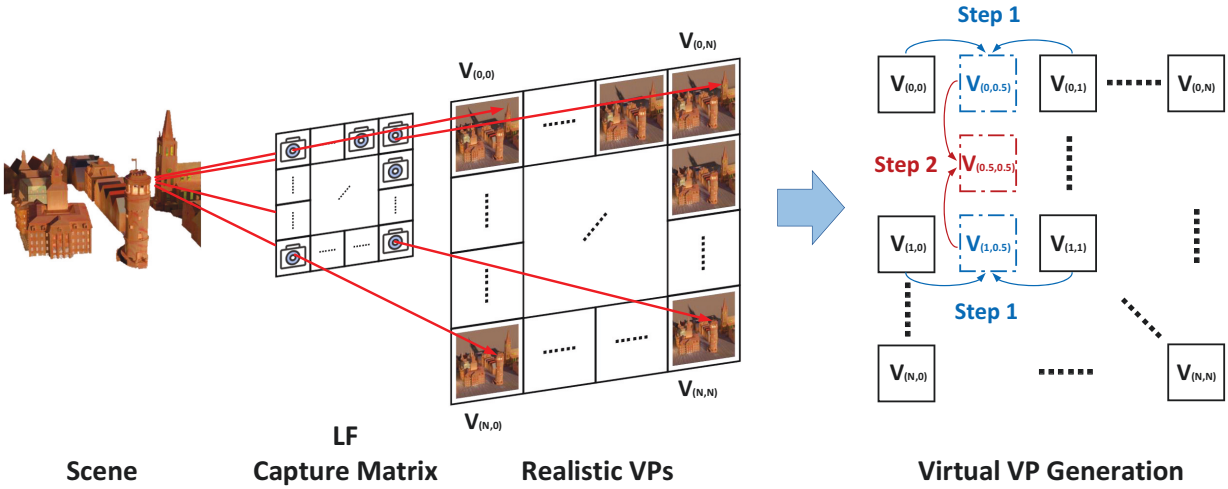


Figure 6.1: Light field acquisition and virtual viewpoint generation. (Reprinted from [J2], submitted to IEEE for peer review.)

To solve the above problem, an edge computing framework is introduced for optimal caching and computing scheduling. The proposed framework establishes an LF delivery model between the edge node (EN) and the user's equipment (UE). Figure 6.2 shows the architecture of this framework, which consists of an edge server node, e.g., base station, WiFi access point, and multiple users using LF devices, e.g., head-mounted display. The VPs cached on the UE mainly come from the user's past navigation records and will remain in the user's cache until overwritten. The cached VPs on the EN are mainly composed of the actual VPs originally collected and some virtual VPs generated in advance. As previously mentioned, caching pregenerated virtual VPs can reduce the computational load and latency for generating a virtual VP. It is assumed that the scheduling of all UEs is synchronized, and all UE's offloading strategy and resource allocation scheme will be periodically rescheduled at a fixed temporal interval.

For the architecture shown in Figure 6.2, the simplest case is for the requested VP to be in the EN or UE cache, as there is no need to generate a new VP. A more complicated case occurs when a virtual VP, which can be generated by both the UE and EN, is requested. One problem here is that if the virtual VP is generated on the EN and sent to the UE, more communication overhead will be incurred. In contrast, if the virtual VP is generated

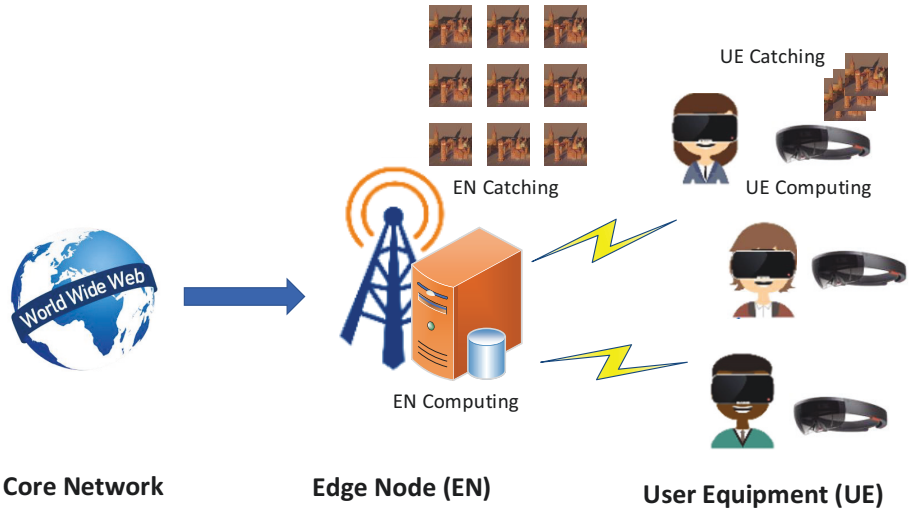


Figure 6.2: Edge-based LF delivery model. (Reprinted from [J2], submitted to IEEE for peer review.)

on the UE, the computing burden on the UE will be heavier. In this work, an optimal scheduling of computing and caching resources is designed for one EN and multiple UE under the constraint of latency. The final optimization goal is to minimize the average computational/transmission energy consumption of UE and the data caching consumption of the EN.

6.2 System Model

Suppose that sets of K UE, which are requesting to watch LF videos, denoted by $U = \{u_1, u_2, ..., u_K\}$, connect to the same nearby edge server. They keep requesting LF VPs from the EN, sharing a limited mobile downlink network channel. The time is divided into discrete time slots $t \in \mathcal{T} = \{1, ..., T\}$, each of which has a duration that matches the timescale at which the user's VP switching request is received. Correspondingly, the offloading configuration and resource allocation brought by the new VP rendering request need to be updated. Let $x_{i,t}$ indicate the offloading configuration for u_i in time slot t. When $x_{i,t}$ is 0, all computing tasks are executed locally on the UE, and when $x_{i,t}$ is 1, all tasks are offloaded to the EN. This work does not consider invoking the computing resources of both the UE and EN simultaneously to complete a single request.

6.2.1 Computing Model

The EN and UE are assumed to run at a given CPU-cycle frequency. The computational frequency of u_i is denoted as f_{u_i} (in $cycle/s$), while the computational frequency of the EN is denoted as $f_{e_{total}}$ (in $cycle/s$), which is limited and will be allocated to different UE requests $f_{e_i}(i = 1, 2, 3..., K)$. If both the EN and UE caches can satisfy u_i's requirement for generating virtual VPs (left/right eyes), r_i^E denotes the number of interpolation rounds required on the EN, and the required interpolation rounds on the UE are represented by r_i^U. The computing cycles required for each round of interpolation are α (in $cycles$). In this work, a realistic assumption is made that the LF resources requested by UEs are different from each other, and cache sharing among UE is not considered.

It should be noted that the proposed model uniformly converts the computing cost of the platform or device into energy consumption and service response latency. The need for the latter is well known: the response time needs to be high so that users do not experience noticeable lags. The former is considered because the LF device is usually powered by battery, and energy consumption becomes important. Ensuring that the UE does not run out of power leads to offloading tasks to the EN. The energy consumed in the computation of one cycle with a computational frequency of f_{u_i} on u_i's mobile LF device is $k_i f_{u_i}^2$, where k_i is the parameter related to the chip architecture and can represent the CPU's power efficiency. The energy consumption for the EN is not considered because it has a continuous power supply. Hence, the average computing energy consumption of all requests in time slot t is:

$$e_t^{comp} = \frac{1}{K} \sum_{i=1}^{K} (1 - x_{i,t}) \times \alpha r_{i,t}^U \times k_i f_{u_i,t}^2 \tag{6.1}$$

The calculation of the latency for each request is decided by the offloading configuration; thus, the average latency of all requests in time slot t is:

$$l_t^{comp} = \frac{1}{K} \sum_{i=1}^{K} (1 - x_{i,t}) \frac{\alpha r_{i,t}^U}{f_{u_i,t}} + x_{i,t} \frac{\alpha r_{i,t}^E}{f_{e_i,t}} \tag{6.2}$$

6.2.2 Communication Model

Since the export bandwidth B_{total} (in Mbps) of the EN is limited, the sum of the throughput Th_i (in Mbps) between the EN and all UE cannot exceed B_{total}. Transmission latency is one

of the important factors in determining resource allocation and offloading configuration. In addition, since the mobile communication module of the UE is usually one of the main factors affecting the energy consumption of the device, the energy consumption caused by data transmission will also be considered. DS_i (in Mbits) denotes the data size of VPs that need to be transmitted between u_i and the EN, which is constrained by the offloading configuration $x_{i,t}$. DS_i is defined as:

$$DS_i = 2S_i \times x_i \tag{6.3}$$

where S_i (in Mbits) is the data size of one single VP of the LF resource requested by u_i.

In the first case of Equation (6.3), when deciding to locally perform VP generation, i.e., $x_i = 0$, the data size of the VP to be transmitted between the EN and the UE is zero. The second case means that the offloading configuration obtains the virtual VP pair from the EN; i.e., $x_i = 1$. Whether they exist directly in the EN cache or need to be generated after interpolation, only two VPs from the EN, which are the virtual pair of VPs, need to be downloaded by the UE. The average transmission latency of all requests in time slot t is:

$$l_t^{trans} = \frac{1}{K} \sum_{i=1}^{K} \frac{DS_{i,t}}{Th_{i,t}} \tag{6.4}$$

Since the main transmission energy consumption of the UE occurs during data transmission, it is assumed that the power of the mobile communication module used for u_i is P_i when it is working and 0 when it is idling. Then, the average transmission energy consumption of all requests in time slot t is:

$$e_t^{trans} = \frac{1}{K} \sum_{i=1}^{K} P_i \times \frac{DS_{i,t}}{Th_{i,t}} \tag{6.5}$$

6.2.3 Caching Model

For LF, caching is mainly composed of two components. The first component consists of all actual VPs (a quantity of $Cach_{i_{Real}}$) and some pregenerated virtual VPs (a quantity of $Cach_{i_{Pre}}$) for each LF resource. Since the proposed model assumes that the LF resources requested by all UEs are different, the average buffer occupied by the first component cache on the EN is:

$$Cach^1 = \frac{1}{K} \sum_{i=1}^{K} (Cach_{i_{Real}} + Cach_{i_{Pre}}) \times S_i \tag{6.6}$$

It should be noted that $Cach^1$ in Equation (6.6) is time-slot invariant. This is mainly because the assumption that the process of pregenerating VPs will only be performed when an LF resource is not requested by the UE. Once a UE requests an LF resource, the $Cach_{i_{Pre}}$ of this LF resource will be locked and cannot be updated. Since $Cach_{i_{Real}}$ does not change with time after being captured by the LF capture matrix, $Cach^1$ is consequently time-slot invariant.

The second component of the cache is the buffer for storing intermediate interpolation results. If the virtual VP requested by u_i needs to be generated through r_i^E rounds of interpolation on the EN, the request needs to temporarily occupy a buffer of size $r_i^E \times S_i$. By accumulating all VP generation tasks offloaded to the EN in time slot t, the average buffer occupied by the second component cache on the EN can be calculated by:

$$Cach_t^2 = \frac{1}{K} \sum_{i=1}^{K} x_{i,t} r_{i,t}^E S_i \tag{6.7}$$

Then, the average EN caching consumption for all requests in time slot t is:

$$Cach_t = Cach^1 + Cach_t^2 \tag{6.8}$$

Since this model assumes that the buffers of all UEs are of a fixed size and will be cyclically used, there is no competition or sharing relationship of cache resources between different UEs; therefore, the usage of UE caches is not considered in this model.

6.2.4 Problem Formulation

The joint optimization problem is formulated to minimize the required average energy and caching consumption under the long-term latency constraint. For simplicity of illustration, the system utility function is defined as the weighted sum of energy and caching consumption. The natural objective is minimizing the time-averaged utility for all UE and the EN,

which can be formulated as:

$$\begin{aligned} \min_{x,Th,f} \quad & \lim_{T\to+\infty} \frac{1}{T} \sum_{t=1}^{T} e_t^{comp} + e_t^{trans} + \omega Cach_t \\ s.t. \quad & C_1 : x_{i,t} = \{0,1\}, u_i \in U, t \in \mathcal{T} \\ & C_2 : \sum_{i=1}^{K} x_{i,t} f_{e_i,t} \leq f_{e_{total}} \\ & C_3 : \sum_{i=1}^{K} x_{i,t} Th_{i,t} \leq B_{total,t} \\ & C_4 : \lim_{T\to+\infty} \frac{1}{T} \sum_{t=1}^{T} l_t^{comp} + l_t^{trans} \leq L_{max} \end{aligned} \tag{6.9}$$

where ω is the weighted parameter used to control the tradeoff between energy and caching consumption. Thus, the optimal solution to this problem trades the average caching consumption on EN to lower the average energy consumption of all UEs. Constraint C_1 ensures that, in each time slot, the computing task of each UE request will only be locally performed or offloaded to the EN. Constraint C_2 states that the sum of the computational frequency occupied by all tasks offloaded to the EN cannot exceed the maximum computational capacity of the EN. The third constraint C_3 ensures that the sum of the throughput used by all requests offloaded to the EN does not exceed the maximum export bandwidth of the EN. The last constraint C_4 requires that the long-term average latency does not exceed the threshold L_{max}.

6.3 Optimization Method

6.3.1 Problem Transformation Using Lyapunov Optimization

Theoretically, the proposed optimization of the previous section is a mixed integer nonlinear programming (MINLP) problem, which is difficult to optimally solve because it is impossible to accurately predict the state of the communication network and the request of each UE in the future. Therefore, the original time-averaged problem is transformed into a series of real-time minimization problems leveraging the Lyapunov optimization [76] to decouple the long-term latency constraint C_4. A virtual queue is defined to measure the value of the total system latency exceeding the maximum tolerated latency, which is given as follows:

$$\begin{aligned} & Q(t+1) = \\ & \quad \max\left\{Q(t) + l_t^{comp} + l_t^{trans} - L_{max}, 0\right\} \end{aligned} \tag{6.10}$$

where $Q(0) = 0$.

The quadratic Lyapunov function $L(Q(t))$ is defined as:

$$L(Q(t)) = \frac{1}{2}Q^2(t) \tag{6.11}$$

The conditional one-slot Lyapunov drift $\Delta(Q(t))$ is defined as:

$$\Delta(Q(t)) = \mathbb{E}[L(Q(t+1)) - L(Q(t))] \tag{6.12}$$

where $L(Q(t))$ represents the scalar measure of latency queue congestion; the smaller it is, the less queue backlog. $\Delta(Q(t))$ reveals the stability of the virtual queue, and its decrease means more stability for the queue. In this work, the goal is to find the best offloading configuration and resource allocation strategies to coordinate the energy consumption of UE and the caching consumption of EN. By incorporating latency queue stability into the tradeoff between energy and caching consumption, a Lyapunov drift-plus-penalty function is defined as:

$$\begin{aligned} \varepsilon(t) =& \Delta(Q(t)) + \\ & V \cdot \mathbb{E}[e_t^{comp} + e_t^{trans} + \omega Cach_t | Q(t)] \end{aligned} \tag{6.13}$$

where $V > 0$ controls the relative significance of minimizing the average latency against reducing the energy and caching consumption during the current time slot.

According to the Lyapunov optimization theorem, the upper bound of the drift-plus-penalty function can be obtained:

$$\begin{aligned} \varepsilon(t) \leq & B + Q(t)\mathbb{E}[l_t^{comp} + l_t^{trans} - L_{max} | Q(t)] \\ & + V \cdot \mathbb{E}[e_t^{comp} + e_t^{trans} + \omega Cach_t | Q(t)] \end{aligned} \tag{6.14}$$

where B is a constant and $B = \frac{1}{2}(l_{max}^2 + L_{max}^2)$. $l_{max} = max_{t \in T} \{l_t^{comp} + l_t^{trans}\}$ represents the largest average latency in all time slots.

Next, the original optimization problem in (Equation (6.9)) is transformed to minimize the supremum bound for the drift-plus-penalty function $\varepsilon(t)$ during each time slot. The new real-time optimization problem can be rewritten as:

$$\begin{aligned} \min_{x,Th,f} \quad & Q(t)(l_t^{comp} + l_t^{trans}) \\ & + V \cdot (e_t^{comp} + e_t^{trans} + \omega Cach_t) \\ s.t. \quad & C_1, C_2, C_3 \end{aligned} \tag{6.15}$$

We can see that the long-term stochastic optimization problem (Equation (6.9)) has

been transformed into a general optimization problem (Equation (6.15)) for a given time slot. The problem (Equation (6.15)) is still an MINLP problem, and its time complexity will exponentially increase as the number of UE increases.

6.3.2 Scheduling Algorithm

In this section, a scheduling algorithm based on the Markov approximation method [77] is proposed to jointly optimize the strategies of offloading configuration and resource allocation. To reduce the complexity of the scheduling algorithm, the optimization problem (Equation (6.15)) can be solved in two steps. In the first step, the optimal EN resource allocation strategy (Th and f) is derived for the given offload configurations (x). In the second step, a search for the optimal offload configurations of all UE is performed.

$X_t = \{x_{i,t} | \forall u_i \in U\}$ is used to denote the collection of all UE offloading configurations. $F_t = \{f_{e_i,t} | \forall u_i \in U\}$ represents the EN computational frequency allocation for all UE requests. $TH_t = \{Th_{i,t} | \forall u_i \in U\}$ is the allocation of the network bandwidth. Suppose X_t is fixed. The problem (Equation (6.15)) can then be reformulated as:

$$\begin{aligned} \min_{TH_t, F_t} \quad & \frac{1}{K} \sum_{i=1}^{K} (Q(t)(x_{i,t} \frac{\alpha r_{i,t}^E}{f_{e_i,t}} + \frac{DS_{i,t}}{Th_{i,t}}) \\ & + V \cdot P_i \frac{DS_{i,t}}{Th_{i,t}}) \\ s.t. \quad & C_2, C_3 \end{aligned} \tag{6.16}$$

Since the terms containing f_{e_i} and $Th_{i,t}$ in the above problem are not coupled with each other, they can be split into two independent optimization problems:

$$\begin{aligned} \min_{TH_t} \quad & \frac{1}{K} \sum_{i=1}^{K} (Q(t) + V \cdot P_i) \frac{2 x_{i,t} S_i}{Th_{i,t}} \\ s.t. \quad & C_3 \end{aligned} \tag{6.17}$$

$$\begin{aligned} \min_{F_t} \quad & \frac{1}{K} \sum_{i=1}^{K} Q(t) x_{i,t} \frac{\alpha r_{i,t}^E}{f_{e_i,t}} \\ s.t. \quad & C_2 \end{aligned} \tag{6.18}$$

For (Equation (6.17)), the constraint C_3 is convex. The objective function in (Equa-

tion (6.17)) is denoted as $obj_1(TH_t)$. The second-order derivatives of obj_1 are $\frac{\partial^2 obj_1}{\partial Th_{i,t}^2} = \frac{2x_{i,t}(Q(t)+V\cdot P_i)S_i}{Th_{i,t}^3} > 0$ and $\forall u_i \in U$. $\frac{\partial^2 obj_1}{\partial Th_{m,t}\partial Th_{n,t}} = 0$, $\forall u_m, u_n \in U, m \neq n$. Hence, the Hessian matrix of obj_1 is positive semidefinite. The problem is convex and satisfies the Karush–Kuhn–Tucker (KKT) condition, so the optimal bandwidth allocation can be derived as follows:

$$Th_{i,t}^* = \frac{B_{total}\sqrt{2x_{i,t}(Q(t)+V\cdot P_i)S_i}}{\sum_{u_i\in U}\sqrt{2x_{i,t}(Q(t)+V\cdot P_i)S_i}} \tag{6.19}$$

Similar to (Equation (6.18)), the optimal allocation for the computational frequency of the EN can be derived as follows:

$$f_{i,t}^* = \frac{f_{e_{total}}\sqrt{x_{i,t}\alpha r_{i,t}^E}}{\sum_{u_i\in U}\sqrt{x_{i,t}\alpha r_{i,t}^E}} \tag{6.20}$$

Based on the above analysis, once the offloading configurations are determined, there is a closed-form optimal solution to the allocation problem of the computing/bandwidth resources. Since the variables of the offloading configuration $x_{i,t}$ are binary, it is impossible to find the optimal solution for the whole problem within a polynomial time. Therefore, a near-optimal searching algorithm for offloading configuration is proposed based on the Markov approximation, as shown in Algorithm 6.1.

The proposed algorithm searches for the optimal offloading configuration X_t for each time slot. For time slot t, as long as the number of iterations does not exceed the maximum steps I_{max} and the objective value θ in Equation (6.16) does not converge, the algorithm will iteratively search for a new feasible offloading configuration $\hat{X}_t$. The condition for judging the convergence of θ is that its value difference $|\hat{\theta}-\theta|$ is smaller than the preset convergence threshold for more than 10 iterations. In each iteration, a random UE u_i changes its offloading configuration $x_{i,t}$ to $\hat{x}_{i,t}$; then, the new optimal bandwidth allocation $\hat{TH}_t^*$ and the computational frequency allocation of the EN $\hat{F}_t^*$ can be derived for all UE according to Equation (6.19) and Equation (6.20). Then, whether to update X_t is decided by the transition probability $\rho \leftarrow \frac{1}{1+e^{\frac{\hat{\theta}-\theta}{\tau}}}$, where τ is the degree of randomness to control the smoothness between exploration and exploitation. A smaller τ will allow the algorithm to more frequently update its offload configuration X_t, but this may cause the algorithm to more easily converge to the local optimum rather than the global optimum. An excessively large τ may make convergence difficult, so a reasonable setting of τ is very important.

Algorithm 6.1 Searching Algorithm for the Optimal Offloading Configuration

Input:

B_{total}, $f_{e_{total}}$, L_{max}, $f_{u_i,t}$, $r^U_{i,t}$, $r^E_{i,t}$, P_i, S_i $\forall u_i \in U, t \in \mathcal{T}$;

Output:

offloading configuration X_t;

initial $Q(0) \leftarrow 0$, $X_t \leftarrow$ random selection;
for $t \in \mathcal{T}$ **do**
  while $iter \leq I_{max}$ and θ do not converge **do**
    Pick a random user u_i and change its offloading configuration from $x_{i,t}$ to $\hat{x}_{i,t}$;
    if $\hat{x}_{i,t}$ is feasible **then**
      $\hat{X}_t \leftarrow \{x_{1,t}, x_{2,t}, ..., \hat{x}_{i,t}, ..., x_{K,t}\}$;
      for $u_i \in U$ **do**
        Obtain $\hat{Th}^*_{i,t}$ according to Equation (6.19);
        Obtain $\hat{f}^*_{i,t}$ according to Equation (6.20);
      end for
      Transition probability $\rho \leftarrow \frac{1}{1+e^{\frac{\hat{\theta}-\theta}{\tau}}}$;
      With ρ, X_t is updated to $\hat{X}_t$;
      With $(1-\rho)$, X_t is unchanged;
    else
      X_t is unchanged;
    end if
  end while
  Obtain $Q(t+1)$ according to Equation (6.10); **return** X_t;
end for

6.4 Simulation Results

The performance of the proposed scheduling algorithm is evaluated and compared with several benchmarks. The basic parameters of the model are set according to the test results on the author's own LF video dataset [C1], which contains 9 groups of real-world LF videos, with an angular resolution of 10×10 and a spatial resolution of 1920×1056. Assuming that the buffer of each LF video needs to cache 1 minute of LF frames, its real-time storage overhead is approximately 1.5 GB/time slot. The mobile network trace dataset is used from [2] to simulate the export bandwidth of the EN. The bandwidth B_{total} varies from 20 Mbps to 400 Mbps. The parameters in the model are set according to the test results of the VP synthesis method of [56], and the processing latency for generating a new VP is roughly distributed between 40 ms and 250 ms. The maximum latency L_{max} of the objective value is set to 100 ms. The average energy power of the UE for local computing is approximately 4(W), and the power of transmission is set to 1(W).

Figure 6.3 presents the convergence performance of the proposed scheduling algorithm in the iterative process. When τ is large (e.g., $\tau = 0.5$), the objective value greatly fluctu-

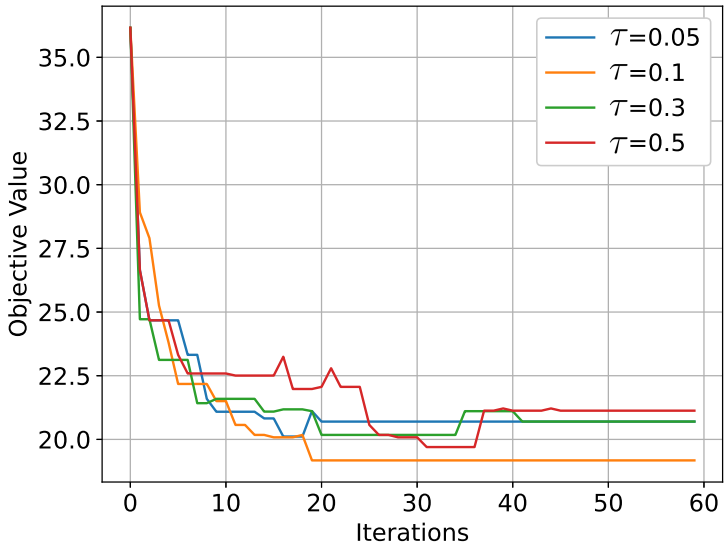


Figure 6.3: Convergence of the scheduling algorithm. (Reprinted from [J2], submitted to IEEE for peer review.)

ates during the algorithm search process, which is not conducive to algorithm convergence. When τ is small (e.g., $\tau = 0.05$), the algorithm will become trapped in a local optimal

solution, and the overall system utility cannot be maximized. In this experiment, the most appropriate value of τ is 0.1, from which the best balance between convergence smoothness and search result quality is achieved.

Figure 6.4 to Figure 6.6 show the simulation results of a set of real bandwidth data for 80 consecutive time slots.

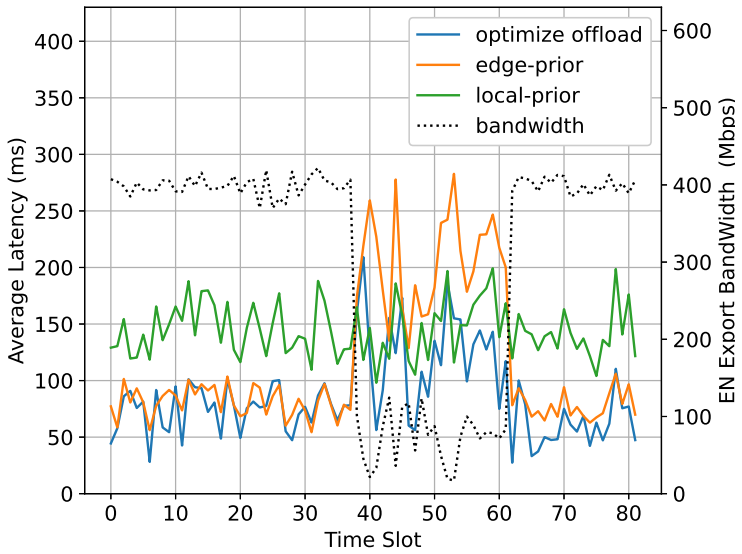


Figure 6.4: Latency comparison for different offloading strategies. (Reprinted from [J2], submitted to IEEE for peer review.)

Most of the time, the total export bandwidth of the EN is stable at approximately 400 Mbps; however, a period of continuous attenuation and oscillation occurs between time slots 40 and 60, which may be caused by sudden changes in the channel environment. In this simulation, three different offloading strategies are compared with each other: **edge-prior** offloads all VP generation tasks to the EN, and there is no computational load on the UE; **local-prior** locally performs all VP generation tasks as much as possible; however, since the local cache of some UE may not meet the requirements for generating the virtual VP, their request tasks will be forcibly offloaded to the EN; **optimize offload** represents the proposed algorithm.

Figure 6.4 to Figure 6.6 show the performance of the response latency, average UE energy consumption, and EN caching consumption at runtime. It can be seen from Figure 6.4 that the latency after using the scheduling algorithm is better than that of the other two benchmarks most of the time. Due to the limited computational capability of the UE, the

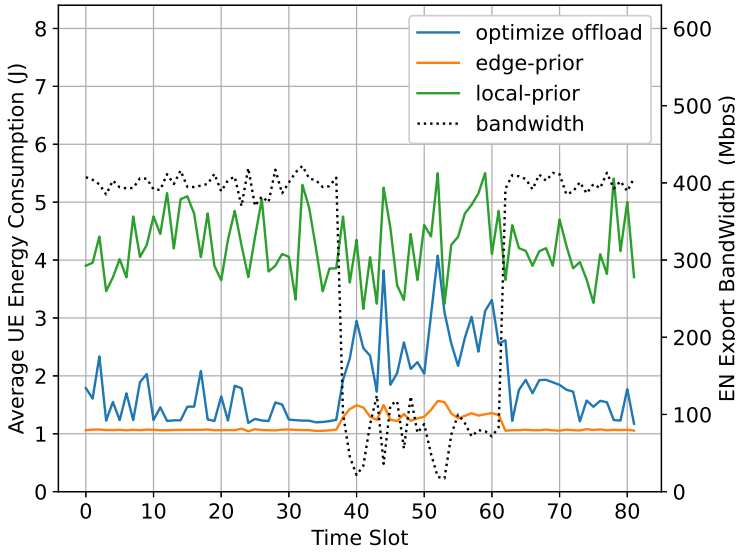


Figure 6.5: Energy consumption comparison for different offloading strategies. (Reprinted from [J2], submitted to IEEE for peer review.)

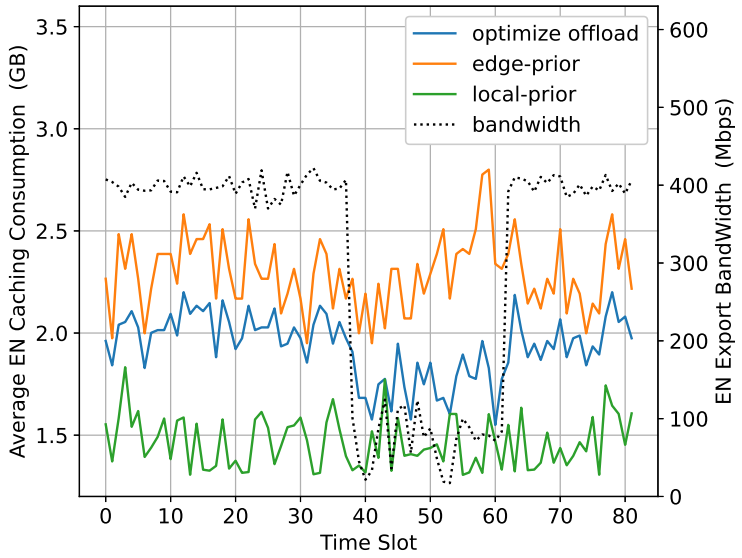


Figure 6.6: Caching consumption comparison for different offloading strategies. (Reprinted from [J2], submitted to IEEE for peer review.)

local-prior strategy will cause significantly higher processing latency than the other two strategies, and the long-term heavy load also causes the UE energy consumption in Figure 6.5 to be higher. However, **local-prior** is friendlier to caching resources on the EN and has the least dependence on network conditions, so its performance suffers the least when the bandwidth decreases. When the bandwidth is sufficient, the strategy adopted by the scheduling algorithm is more similar to **edge-prior**, and it is more likely to offload tasks to the EN. On the one hand, a tradeoff between the energy consumption of the UE and the caching consumption of the EN can be achieved, thereby reducing the burden on the UE. On the other hand, the processing latency of some high-computational tasks can be reduced at the expense of a smaller transmission latency; i.e., a tradeoff between computing and communication. However, when the bandwidth decreases, the communication latency will increase and eventually exceed the reduced processing latency, no longer guaranteeing the overall latency constraint. At this time, the scheduling algorithm will tend to reserve more computing tasks for local execution and reduce the burden on the EN by exchanging communication for computing. We can see from Figure 6.5 and Figure 6.6 that while the UE energy consumption increases, the EN caching consumption decreases, which indicates that the system resource allocation of the proposed scheduling algorithm is comprehensive and adaptive. In summary, the proposed scheduling algorithm can dynamically adjust the offloading configuration and resource allocation strategy according to changes in network conditions to achieve a balance between user energy consumption and edge caching consumption and simultaneously provide users with the best experience in terms of response latency.

As previously mentioned, increasing the number of VPs cached on the EN ($Cach_{i_{Real}} + Cach_{i_{Pre}}$) can reduce the rounds of interpolation required to generate new VPs, thereby reducing the overall latency. Figure 6.7 shows the impact on the average latency and caching consumption when VPs of different densities are stored in advance on the EN. We can see that the reduction in user response latency comes at the expense of an exponential increase in the EN caching consumption. Although we do not consider the scheduling and optimization of $Cach_{i_{Real}} + Cach_{i_{Pre}}$ in this model, it is necessary to set a feasible VP prestorage density on EN, which is 9×9 in this work.

In the last experiment, the impact of the number of users is tested, as shown in Figure 6.8. When the number of users is small, the bandwidth and computational frequency that can be allocated to one request of a UE are abundant. At this time, the benefit of offloading tasks from the UE to the EN is substantial for the scheduling algorithm, so the ratio of tasks offloaded to the EN is very high. However, as the number of users increases, the EN resources become less available, and offloading to the EN will no longer be beneficial; thus, the scheduling algorithm arranges for more tasks to be locally executed. When

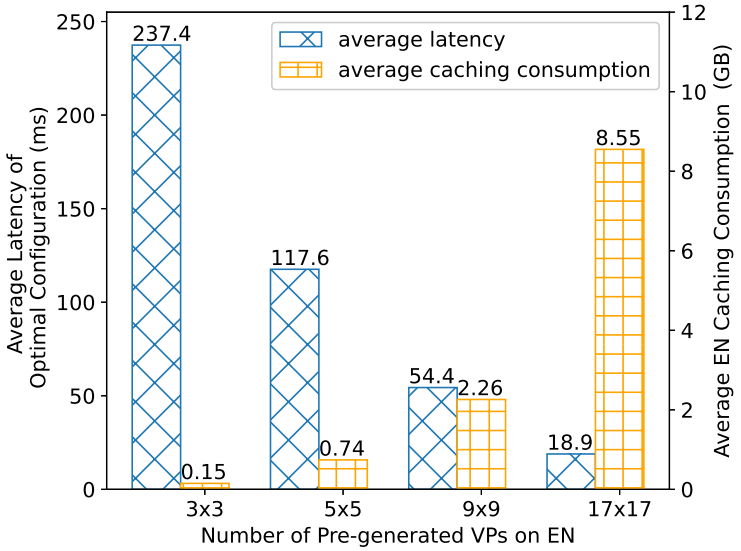


Figure 6.7: Impact of the prestorage VP density on the EN: $Cach_{i_{Real}} + Cach_{i_{Pre}}$. (Reprinted from [J2], submitted to IEEE for peer review.)

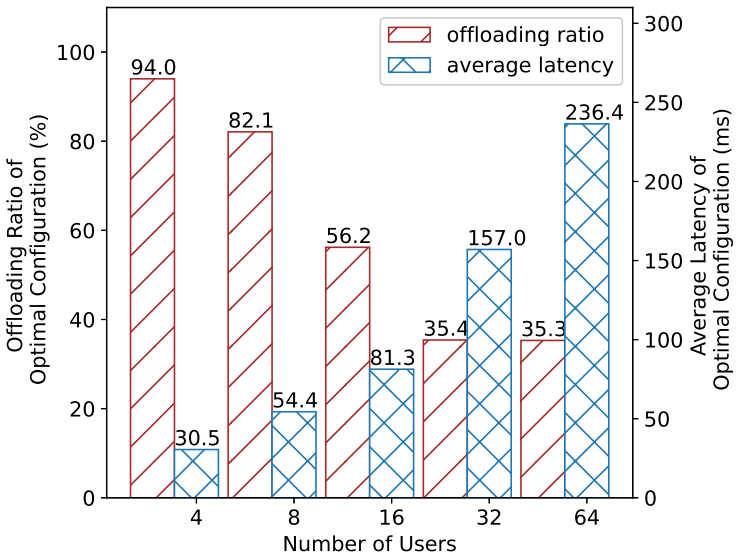


Figure 6.8: Impact of the number of users K. (Reprinted from [J2], submitted to IEEE for peer review.)

the number of users is 64, the offloading ratio no longer decreases because the remaining nearly 35% of the tasks can only be executed in the EN. The local cache of the UE is not sufficient to support the generation of the requested VP. Although the maximum latency L_{max} set in advance cannot be satisfied when the number of users is very large, the scheduling algorithm still attempts to maximize utility. This shows that the proposed method has good stability and can still work under excessive pressure.

6.5 Chapter Summary

In this chapter, an edge computing system for immersive LF VP generation is proposed, the system is modeled, and an optimal scheduling algorithm is designed. This algorithm can adaptively optimize the offload configuration and resource allocation according to the energy consumption of users' devices and the caching consumption of the edge server under the constraint of long-term response latency. Experimental results show that the edge-computing architecture is beneficial for LF VP generation. The proposed algorithm can not only achieve load balancing between local devices and an edge server but also decides on the tradeoffs between computing, communication and caching well.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

In this thesis, the challenges faced by LF transmission and processing technology are explored and discussed from multiple perspectives, and corresponding solutions are proposed. A variety of novel applied AI methods (GNN and CNN) and communication technologies (MDC and MEC) have been proven to be capable of addressing different LF technology bottlenecks well.

The main focus of Chapter 3 is the design of a more efficient and stable LF compression scheme. This shows that the idea of using a graph model to represent the correlation between viewpoints and using a GNN for viewpoint reconstruction works very well in LF compression. The two-layer compression architecture of SAI image frequency component separation effectively solves the performance stability problem of the GNN-based reconstruction method on different LF data so that the overall compression scheme can be widely used in practical applications.

In Chapter 4, a discussion of whether LF streaming can still work reliably in under-resourced and error-prone network environments is presented. This discussion differs from the one in Chapter 3, which only covers the compression of LF video data sources and does not need to consider the impact of fluctuations in the network environment during LF transmission. However, in Chapter 4, an application layer protocol for LF streaming that controls and adjusts the entire LF transmission process in real time is described.

In Chapter 5, the focus is on LF depth estimation, a classic LF processing and application problem. The goal is to not only design different estimation methods for wide-

and narrow-baseline LF but also to uniformly process LF data of different baselines using optical flow for support. This enables the proposed depth estimation algorithm to have better adaptability to different LF data sources in practice.

In Chapter 6, the joint design of LF transmission and processing is discussed, and the possibilities for collaborative improvement between communication and computing are explored. This proves that MEC technology has great application potential in the LF area.

7.2 Recommendations for Future Work

The author has performed some research in the area of LF processing and transmission, following the current mainstream research direction of LF. However, the author believes that the development of LF technology is more in need of research breakthroughs in terms of the usage (or application) method and acquisition method of LF data.

For the usage method, it is obviously not of much value to simply provide users with many viewpoints for them to choose at will. The author believes that converting LF into a 3D model and rendering it to the user is a more valuable LF data application method that can be foreseen at present. Using the central SAI for texture mapping after depth estimation is a relatively mature and direct implementation. However, texture and depth mismatching is likely to have a large impact on the precision of the produced 3D models. Exploring a novel way to build a 3D scene model directly from the SAI matrix is a very promising research direction.

For the acquisition method, an increasing number of researchers have noticed that although the LF enables the user's 6 DOF experience, the array acquisition system with most of the lenses facing the same direction makes the LF lose the ability to fully cover 360-degrees of the scene. Therefore, due to the existing acquisition methods, LF actually only has a theoretical immersive media function. Google has already begun work in this area, designing spherical LF acquisition arrays in recent years, as shown in Chapter 2. In addition, the author believes that it may also be a feasible solution to assemble multiple Lytro lenslet cameras like traditional 360 panoramic cameras. Many techniques and methods for 360-degree video can be directly applied or referenced. Designing the new 360-degree LF cameras is another suggested research direction of this thesis.

References

[1] Lixin Shi, Haitham Hassanieh, Abe Davis, Dina Katabi, and Fredo Durand. Light field reconstruction using sparsity in the continuous fourier domain. *ACM Transactions on Graphics (TOG)*, 34(1):12, 2014.

[2] Darijo Raca, Jason J Quinlan, Ahmed H Zahran, and Cormac J Sreenan. Beyond throughput: a 4g lte dataset with channel and context metrics. In *Proceedings of the 9th ACM Multimedia Systems Conference*, pages 460–465, 2018.

[3] J. van der Hooft, S. Petrangeli, T. Wauters, R. Huysegems, P. R. Alface, T. Bostoen, and F. De Turck. HTTP/2-Based Adaptive Streaming of HEVC Video Over 4G/LTE Networks. *IEEE Communications Letters*, 20(11):2177–2180, 2016.

[4] Wei Xiang, Gengkun Wang, Mark Pickering, and Yongbing Zhang. Big video data for light-field-based 3D telemedicine. *IEEE Network*, 30(3):30–38, 2016.

[5] Daniel Gotsch, Xujing Zhang, Timothy Merritt, and Roel Vertegaal. Telehuman2: A cylindrical light field teleconferencing system for life-size 3D human telepresence. In *CHI Conference on Human Factors in Computing Systems*, page 1–10, 2018.

[6] Ryan S Overbeck, Daniel Erickson, Daniel Evangelakos, Matt Pharr, and Paul Debevec. A system for acquiring, processing, and rendering panoramic light field stills for virtual reality. *ACM Transactions on Graphics (TOG)*, 37(6):1–15, 2018.

[7] Li Li, Zhu Li, Bin Li, Dong Liu, and Houqiang Li. Pseudo-sequence-based 2-d hierarchical coding structure for light-field image compression. *IEEE Journal of Selected Topics in Signal Processing*, 11(7):1107–1119, 2017.

[8] Caroline Conti, Luís Ducla Soares, and Paulo Nunes. Hevc-based 3D holoscopic video coding using self-similarity compensated prediction. *Signal Processing Image Communication*, 42:59–78, 2016.

[9] Yun Li, Märten Sjöström, Roger Olsson, and Ulf Jennehag. Efficient intra prediction scheme for light field image compression. In *2014 IEEE International conference on acoustics, speech and signal processing (ICASSP)*, pages 539–543. IEEE, 2014.

[10] Gaochang Wu, Belen Masia, Adrian Jarabo, Yuchen Zhang, Liangyong Wang, Qionghai Dai, Tianyou Chai, and Yebin Liu. Light field image processing: An overview. *IEEE Journal of Selected Topics in Signal Processing*, 11(7):926–954, 2017.

[11] Mohammad Kazemi, Mohammad Ghanbari, and Shervin Shirmohammadi. A review of temporal video error concealment techniques and their suitability for HEVC and VVC. *Multimedia Tools and Applications*, 80(8):12685–12730, 2021.

[12] Michael Broxton, John Flynn, Ryan Overbeck, Daniel Erickson, Peter Hedman, Matthew Duvall, Jason Dourgarian, Jay Busch, Matt Whalen, and Paul Debevec. Immersive light field video with a layered mesh representation. *ACM Transactions on Graphics (TOG)*, 39(4):86–1, 2020.

[13] Satoshi Fujigaki, Kazuya Kodama, and Takayuki Hamamoto. Multi-view imaging system using paraboloidal mirror arrays for efficient acquisition of dynamic light fields. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 3532–3536. IEEE, 2019.

[14] Titus Leistner, Hendrik Schilling, Radek Mackowiak, Stefan Gumhold, and Carsten Rother. Learning to think outside the box: Wide-baseline light field depth estimation with epi-shift. In *2019 International Conference on 3D Vision (3DV)*, pages 249–257. IEEE, 2019.

[15] Xiaoran Jiang, Mikaël Le Pendu, and Christine Guillemot. Depth estimation with occlusion handling from a sparse set of light field views. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 634–638. IEEE, 2018.

[16] Jinglei Shi, Xiaoran Jiang, and Christine Guillemot. A framework for learning depth from a flexible subset of dense and sparse light field views. *IEEE Transactions on Image Processing*, 28(12):5867–5880, 2019.

[17] Aleksandra Chuchvara, Attila Barsi, and Atanas Gotchev. Fast and accurate depth estimation from sparse light fields. *IEEE Transactions on Image Processing*, 29:2492–2506, 2019.

[18] Mohammad Kazemi, Shervin Shirmohammadi, and Khosrow Haj Sadeghi. A review of multiple description coding techniques for error-resilient video delivery. *Multimedia Systems*, 20(3):283–309, 2014.

[19] Florian Niedermeier, Gergo Lovasz, and Hermann De Meer. Quantifying IT energy efficiency. *Advances in Computers*, 87:55–87, 2012.

[20] Dinh Trieu Duong. Wyner-ziv scheme based multiple description coding for robust video transmission over path diversity networks. In *2019 International Conference on Advanced Technologies for Communications (ATC)*, pages 137–141. IEEE, 2019.

[21] E. H. ADELSON. The plenoptic function and the elements of early vision. *Computational Models of Visual Processing*, 1991.

[22] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31–42, 1996.

[23] Ren Ng, Marc Levoy, Mathieu Brédif, Gene Duval, Mark Horowitz, Pat Hanrahan, et al. Light field photography with a hand-held plenoptic camera. *Computer Science Technical Report CSTR*, 2(11):1–11, 2005.

[24] Amar Aggoun. Compression of 3D integral images using 3D wavelet transform. *Journal of Display Technology*, 7(11):586–592, 2011.

[25] Dong Liu, Lizhi Wang, Li Li, Zhiwei Xiong, Feng Wu, and Wenjun Zeng. Pseudo-sequence-based light field image compression. In *IEEE International Conference on Multimedia and Expo Workshops*, pages 1–4, 2016.

[26] Luís F. R. Lucas, Caroline Conti, Paulo Nunes, Luís Ducla Soares, Nuno M. M. Rodrigues, Carla L. Pagliari, Eduardo A. B. Da Silva, and Sérgio M. M. De Faria. Locally linear embedding-based prediction for 3D holoscopic image coding using HEVC. In *Signal Processing Conference*, pages 11–15, 2014.

[27] Ricardo Monteiro, Luis Lucas, Caroline Conti, Paulo Nunes, Nuno Rodrigues, Sergio Faria, Carla Pagliari, Eduardo Da Silva, and Luis Soares. Light field HEVC-based image coding using locally linear embedding and self-similarity compensated prediction. In *IEEE International Conference on Multimedia and Expo Workshops*, pages 1–4, 2016.

[28] Cristian Perra and Daniele Giusto. Jpeg 2000 compression of unfocused light field images based on lenslet array slicing. In *IEEE International Conference on Consumer Electronics*, pages 27–28, 2017.

[29] Xiang Zhang, Philip A. Chou, Ming-Ting Sun, Maolong Tang, Shanshe Wang, Siwei Ma, and Wen Gao. Surface light field compression using a point cloud codec. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 9(1):163–176, 2019.

[30] C. Perra and P. Assuncao. High efficiency coding of light field images based on tiling and pseudo-temporal data arrangement. In *IEEE International Conference on Multimedia and Expo Workshops*, pages 1–4, 2016.

[31] Wei Zhang, Dong Liu, Zhiwei Xiong, and Jizheng Xu. Sift-based adaptive prediction structure for light field compression. In *Visual Communications and Image Processing*, pages 1–4, 2018.

[32] Bichuan Guo, Yuxing Han, and Jiangtao Wen. Convex optimization based bit allocation for light field compression under weighting and consistency constraints. In *2018 Data Compression Conference*, pages 107–116, 2018.

[33] H. Amirpour, M. Pereira, and A. Pinheiro. High efficient snake order pseudo-sequence based light field image compression. In *2018 Data Compression Conference*, pages 397–397, March 2018.

[34] Harini Priyadarshini Hariharan, Tobias Lange, and Thorsten Herfet. Low complexity light field compression based on pseudo-temporal circular sequencing. In *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*, pages 1–5, 2017.

[35] D. G. Dansereau, O. Pizarro, and S. B. Williams. Decoding, calibration and rectification for lenselet-based plenoptic cameras. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1027–1034, June 2013.

[36] Shinjini Kundu. Light field compression using homography and 2d warping. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1349–1352, 2012.

[37] Xiaoran Jiang, Mikaël Le Pendu, Reuben A. Farrugia, and Christine Guillemot. Light field compression with homography-based low rank approximation. *IEEE Journal of Selected Topics in Signal Processing*, PP(99):1–1, 2017.

[38] Shan Xu, Zhi-Liang Zhou, and Nicholas Devaney. Multi-view image restoration from plenoptic raw images. In *Asian Conference on Computer Vision*, pages 3–15. Springer, 2014.

[39] Fatma Hawary, Christine Guillemot, Dominique Thoreau, and Guillaume Boisson. Scalable light field compression scheme using sparse reconstruction and restoration. In *Image Processing (ICIP), 2017 IEEE International Conference on*, pages 3250–3254. IEEE, 2017.

[40] Irene Viola, Hermina Petric Maretic, Pascal Frossard, and Touradj Ebrahimi. A graph learning approach for light field image compression. *Applications of Digital Image Processing XLI*, (Spie-Int Soc Optical Engineering):12, 2018.

[41] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *International conference on machine learning*, pages 2014–2023. PMLR, 2016.

[42] Mohammad Kazemi, Razib Iqbal, and Shervin Shirmohammadi. Redundancy allocation based on the weighted mismatch-rate slope for multiple description video coding. *IEEE Transactions on Multimedia*, 19(1):54–66, 2016.

[43] Mohammad Kazemi, Razib Iqbal, and Shervin Shirmohammadi. Joint intra and multiple description coding for packet loss resilient video transmission. *IEEE Transactions on Multimedia*, 20(4):781–795, 2017.

[44] Chunyu Lin, Yao Zhao, Jimin Xiao, and Tammam Tillo. Region-based multiple description coding for multiview video plus depth video. *IEEE Transactions on Multimedia*, 20(5):1209–1223, 2017.

[45] Shyi-Chyi Cheng, Ting-Lan Lin, and Ping-Yuan Tseng. K-svd based point cloud coding for rgb-d video compression using 3D super-point clustering. In *International Conference on Multimedia Modeling*, pages 690–701. Springer, 2020.

[46] Jing Chen, Jie Liao, Huanqiang Zeng, Canhui Cai, and Kai-Kuang Ma. An efficient multiple description coding for multi-view video based on the correlation of spatial polyphase transformed subsequences. *Journal of Imaging Science and Technology*, 63(5):50401–1, 2019.

[47] Ole Johannsen, Antonin Sulc, and Bastian Goldluecke. What sparse light field coding reveals about scene structure. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3262–3270, 2016.

[48] Ting-Chun Wang, Jun-Yan Zhu, Ebi Hiroaki, Manmohan Chandraker, Alexei A Efros, and Ravi Ramamoorthi. A 4d light-field dataset and cnn architectures for material

recognition. In *European conference on computer vision*, pages 121–138. Springer, 2016.

[49] Hae-Gon Jeon, Jaesik Park, Gyeongmin Choe, Jinsun Park, Yunsu Bok, Yu-Wing Tai, and In So Kweon. Accurate depth map estimation from a lenslet light field camera. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1547–1555, 2015.

[50] Shuo Zhang, Hao Sheng, Chao Li, Jun Zhang, and Zhang Xiong. Robust depth estimation for light field via spinning parallelogram operator. *Computer Vision and Image Understanding*, 145:148–159, 2016.

[51] Hao Sheng, Pan Zhao, Shuo Zhang, Jun Zhang, and Da Yang. Occlusion-aware depth estimation for light field using multi-orientation epis. *Pattern Recognition*, 74:587–599, 2018.

[52] Changha Shin, Hae-Gon Jeon, Youngjin Yoon, In So Kweon, and Seon Joo Kim. Epinet: A fully-convolutional neural network using epipolar geometry for depth from light field images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4748–4757, 2018.

[53] Kunyuan Li, Jun Zhang, Rui Sun, Xudong Zhang, and Jun Gao. Epi-based oriented relation networks for light field depth estimation. *arXiv preprint arXiv:2007.04538*, 2020.

[54] Dingcheng Yue, Muhammad Shahzeb Khan Gul, Michel Bätz, Joachim Keinert, and Rafał Mantiuk. A benchmark of light field view interpolation methods. In *2020 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pages 1–6. IEEE, 2020.

[55] Suren Vagharshakyan, Robert Bregovic, and Atanas Gotchev. Light field reconstruction using shearlet transform. *IEEE transactions on pattern analysis and machine intelligence*, 40(1):133–147, 2017.

[56] Eric Penner and Li Zhang. Soft 3D reconstruction for view synthesis. *ACM Transactions on Graphics (TOG)*, 36(6):1–11, 2017.

[57] Can Wang, Sheng Zhang, Yu Chen, Zhuzhong Qian, Jie Wu, and Mingjun Xiao. Joint configuration adaptation and bandwidth allocation for edge-based real-time video analytics. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 257–266. IEEE, 2020.

[58] Yaping Sun, Zhiyong Chen, Meixia Tao, and Hui Liu. Communications, caching, and computing for mobile virtual reality: Modeling and tradeoff. *IEEE Transactions on Communications*, 67(11):7573–7586, 2019.

[59] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

[60] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1024–1034, 2017.

[61] Reza Rassool. Vmaf reproducibility: Validating a perceptual practical video quality metric. In *IEEE International Symposium on Broadband Multimedia Systems and Broadcasting*, 2017.

[62] Stanford University Computer Graphics Laboratory. Light field datasets. http://lightfield.stanford.edu/lfs.html, 2008.

[63] Edwin O Elliott. Estimates of error rates for codes on burst-noise channels. *The Bell System Technical Journal*, 42(5):1977–1997, 1963.

[64] Edgar N Gilbert. Capacity of a burst-noise channel. *Bell system technical journal*, 39(5):1253–1265, 1960.

[65] Ting-Chun Wang, Jun-Yan Zhu, Nima Khademi Kalantari, Alexei A Efros, and Ravi Ramamoorthi. Light field video capture using a learning-based hybrid imaging system. *ACM Transactions on Graphics (TOG)*, 36(4):1–13, 2017.

[66] Laurent Guillo, Xiaoran Jiang, Gauthier Lafruit, and Christine Guillemot. Light field video dataset captured by a r8 raytrix camera (with disparity maps). *International Organisation for Standardisation ISO/IEC JTC1/SC29/WG1 and WG11*, 2018.

[67] Ravi Netravali, Anirudh Sivaraman, Somak Das, Ameesh Goyal, Keith Winstein, James Mickens, and Hari Balakrishnan. Mahimahi: Accurate record-and-replay for HTTP. In *USENIX Annual Technical Conference*, pages 417–429, 2015.

[68] Lijun Zhao, Huihui Bai, Anhong Wang, and Yao Zhao. Multiple description convolutional neural networks for image compression. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(8):2494–2508, 2018.

[69] Lili Meng, Jie Liang, Upul Samarawickrama, Yao Zhao, Huihui Bai, and André Kaup. Multiple description coding with randomly and uniformly offset quantizers. *IEEE Transactions on Image Processing*, 23(2):582–595, 2013.

[70] Sorina Dumitrescu and Xiaolin Wu. Optimal two-description scalar quantizer design. *Algorithmica*, 41(4):269–287, 2005.

[71] Tammam Tillo and Gabriella Olmo. A novel multiple description coding scheme compatible with the jpeg2000 decoder. *IEEE Signal Processing Letters*, 11(11):908–911, 2004.

[72] Huihui Bai, Ce Zhu, and Yao Zhao. Optimized multiple description lattice vector quantization for wavelet image coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(7):912–917, 2007.

[73] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020.

[74] Katrin Honauer, Ole Johannsen, Daniel Kondermann, and Bastian Goldluecke. A dataset and evaluation methodology for depth estimation on 4d light fields. In *Asian Conference on Computer Vision*, pages 19–34. Springer, 2016.

[75] T Tieleman and G Hinton. Divide the gradient by a running average of its recent magnitude. Technical report, Coursera: Neural networks for machine learning, 2017.

[76] Michael J Neely. Stochastic network optimization with application to communication and queueing systems. *Synthesis Lectures on Communication Networks*, 3(1):1–211, 2010.

[77] J Andrés Christen and Colin Fox. Markov chain monte carlo using an approximation. *Journal of Computational and Graphical statistics*, 14(4):795–810, 2005.