TOWARDS EFFICIENT AND ROBUST VOLUMETRIC VIDEO STREAMING

LAI WEI

PhD

The Hong Kong Polytechnic University

2024

The Hong Kong Polytechnic University
Department of Computing

Towards efficient and robust volumetric video streaming

Lai Wei

A thesis submitted in partial fulfillment of the requirements for
the degree of Doctor of Philosophy
July 2024

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

Signature: _____

Name of Student: Lai Wei

Abstract

Immersive video streaming has been drawing escalating attention in recent years and is the foundation of a range of key VR applications, e.g., teleconferencing, remote teaching, sports game broadcasting, and so on. Among many forms of immersive videos, e.g., 360-degree video, Neural Radiance Fields (NeRF), and 3D Gaussian splatting(3DGS), point cloud-based volumetric video is of particular interest due to its good balance between low device dependency, low computation cost, and high immersiveness. Many existing works focus on improving bandwidth efficiency and adaptiveness under local networks but do not discuss the challenges facing the public Internet. This difference could introduce a more stringent bandwidth constraint, vulnerability to security attacks, and diversified viewing conditions. Therefore, it is necessary to propose new schemes and methods to address these new challenges. In this thesis, we conduct an in-depth study of these new problems and make the following original contributions.

Firstly, we propose a bandwidth-efficient volumetric video streaming framework **VSAS** that, for the first time, allows DASH-based video streaming of MPEG V-PCC formatted volumetric video streaming. MPEG V-PCC is a new standard for volumetric video compression, featuring a high compression ratio and effective temporal prediction. However, it is not readily applicable to work with dynamic network environment streaming. First, there is a need for a rate-distortion model for MPEG V-PCC, which is essential for achieving effective bitrate control. Therefore, we conducted one of the

earliest rate-distortion studies on MPEG V-PCC and proposed a geometry-aware model that achieves high accuracy; second, we designed a transformer-based offline reinforcement learning method to control the Bitrate according to the network dynamics and user movements; third, as the coarse-grained DASH architecture causes frequent frame freezing, we propose a DAG-based frame dropping mechanism that enables the existing system with a frame rate scaling capability. Together, our VSAS framework delivers a smooth Internet volumetric video streaming service. Extensive experiments reveal that VSAS has achieved a lower stalling effect, better bandwidth efficiency, and higher visual quality than existing systems.

Secondly, we study the generalization problem in tile-based volumetric video streaming systems and propose **FewVV**. We first identified the limitation of the existing system when facing an out-of-distribution environment, which essentially constrains the real-world deployment of the tile pruning-based optimization of these systems. To tackle this challenge, we noticed the few-shot and zero-shot adaptation ability of the large language models; therefore, we first reformulate the volumetric video streaming control into a multi-variate sequence modeling problem, then train a causal transformer model with prompt-tuning to solve it. Our evaluation demonstrates a consistent improvement compared to several baselines regarding the QoE and the adaptation speed to an unseen environment.

Finally, we study the error concealing problem of volumetric video and built a novel dataset, **VVCorupt**. We first introduce the background of the existing error-concealing algorithms and related datasets, then we identify a lack of existing dataset for training and bench marking the error concealing algorithms. We build a corruption model for the volumetric video streaming according to the network models, and then build a large scale error concealing dataset with reference frame. We analyze the corruption patterns in our collected dataset and point out the potential directions for building an effective error concealing models for volumetric videos.

In summary, we conducted an in-depth study on three major challenges (efficiency,

privacy, and generalization) toward a better volumetric video streaming system and proposed effective methods to tackle them. We evaluate our methods by evaluating prototype systems over various conditions to confirm their applicability. At the end of the thesis, we reveal several insights for future research.

Publications Arising from the Thesis

1. **Lai Wei**, Yanting Liu, Fangxin Wang, Dayou Zhang, and Dan Wang, "VSAS: Decision Transformer-Based On-Demand Volumetric Video Streaming With Passive Frame Dropping", *IEEE Internet of Things Journal (IOTJ)*, December, 2023.

2. **Lai Wei**, Yanting Liu, Fangxin Wang, and Dan Wang, "FewVV: Few-shot Adaptive Bitrate Volumetric Video Streaming with Prompted Online Adaptation", *IEEE Internet of Things Journal (IOTJ)*, October, 2023.

3. Rui Lu*, **Lai Wei***, Shuntao Zhu, Chuang Hu, and Dan Wang, "Pagoda: Privacy Protection for Volumetric Video Streaming through Poisson Diffusion Model", *Proceedings of the 31st ACM International Conference on Multimedia (ACM MM)*, Ottawa ON, Canada, October, 2023. (*:co-first author)

4. Dayou Zhang, **Lai Wei**, Kai Shen, Hao Zhu, Dan Wang, and Fangxin Wang, "TrimStream: Adaptive Realtime Video Streaming in Adverse Network Conditions", *IEEE Transactions on Mobile Computing (TMC)*, December, 2024.

5. **Lai Wei**, Dayou Zhang, Yanting Liu, Fangxin Wang, and Dan Wang, "A Tree-based Volumetric Video Corruption Dataset for Partial Reliable Error Concealing", submitted to *The 32nd IEEE Conference on Virtual Reality and 3D User*

Interfaces (IEEE VR), 2025.

6. Dayou Zhang, Zhicheng Liang, Zijian Cao, **Lai Wei**, Dan Wang, Fangxin Wang, "3DGStream: Towards Efficient 3D Gaussian Splatting Streaming", submitted to *The 44th IEEE International Conference on Computer Communications (INFOCOM)* , 2025.

Acknowledgments

Although the words may never express the whole feelings of my heart, I still want to dedicate my appreciation to the everyone who helped me through my PhD study. Without their help, I would not have reached this stage.

First, I want to give my deepest thanks to my PhD supervisor, Prof. Dan Wang. He encourages me to step onto this inspiring and exciting journey of scientific research. Along this way, he has guided me with significant time and thought, nurtured my decent and rigorous researcher personality, and showed me the direction a research career should take. From an academic perspective, he guided me with critical thinking, careful judgment, and an inspiring sense of the research value. His dedication to research and interest in innovative works have deeply influenced me. It would continuously motivate me to do more. I will benefit from his lessons for my long journey to academic excellence.

I would also like to thank Prof.Chuang Hu, Prof. Fangxin Wang, Prof. Wei Bao, Prof. Tong Li, and Dr. Fanzhao Wang for their discussions, guidance, and inspiring comments during our exciting collaborations. Their different perspectives on the research problems and solutions widened my vision, encouraging me to strive for excellence in my research career. I would like to take them as my role models and work harder to make more meaningful research works.

Furthermore, I thank my most closet group mates, Mr. Yanting Liu, Mr. Yang Deng, Mr. Rui Lu, and Mrs. Siping Shi. Their help and support during my PhD journey

allowed me to cherish this joyful time.

Finally, I give my gratitude and thanks to my family members, especially my wife. It is their constant help and deep faith into my effort give me the strength through this challenging stage of my life.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Immersive video streaming has become a foundation technology component of various VR and XR applications. For example, teleconferencing and telepresence, where holographic [44] communication allows multiple users distributed at different geographical locations to be present in a virtual space. Much research has shown that such a higher immersive presence provides unique advantages over 2D video conferencing, e.g., it provides a faster learning process and helps people to be aware of their emotions. Another example is Sports Live Broadcasting, where the sports game is recorded from multiple views and then forms unified dynamic virtual 3D scenes, which later been streamed over the Internet for viewers to watch on VR headsets, e.g., Oculus Quest 2 [13] and Apple Vision Pro. As the 6G network accelerates, the future of Internet applications could be largely enhanced with immersive video streaming technologies. Multiple immersive video formats exist 360-degree, volumetric, and NeRF [37]. Where 360-degree video [79] is the most accessible format, providing a three-degree-of-freedom (3DoF) experience that allows the viewer to look in different directions (yaw, pitch, row) but cannot allow the viewer to move the origin point of

view, which has largely limited its immersive level. However, with the rise of network bandwidth and the SoC performance of mobile and immersive devices, users demand a higher level of immersiveness. Volumetric video streaming fills in this gap by providing a six-degree-of-freedom (6DoF) experience: three position freedom and three rotation freedom without bringing excessive bandwidth or computation overhead.

Volumetric video was first standardized by MPEG in 2020. Where two standards for volumetric video streaming, MPEG G-PCC [10] and MPEG V-PCC [39], are published. Soon, academia noticed this development, and a range of research emerged to build volumetric video streaming frameworks. This new video format has taken significant steps forward and is now widely supported by commercial VR devices, like Oculus Quest 2 and Apple Vision Pro, as well as 3D cameras like Kinect and Intel RealSense RGB-D cameras. Volumetric videos are comprised of a temporal sequence of frames. Different from 2D frames consisting of pixels arranged in a dense and ordered manner, volumetric video frame is a dense colored point cloud. Each point has its geometry position in the coordinate system and a group of attributes attached to it that describe its visual characteristics, including the color, alpha, and even the norms and point size. These points are unordered sets compared to the ordered pixel in 2D, and these points have an uneven and sparse distribution in a viewing space. These features have opened the design space for an efficient streaming pipeline that effectively utilizes these characteristics. These features then lead to two types of streaming schemes: i) projecting the point clouds to planar atlas maps before streaming the existing 2D video streaming pipeline. ii) The tile-based viewport adaptive streaming depending on pruning the point cloud. Using these schemes, the volumetric video could be streamed over the Internet efficiently and effectively. Providing a basic service to the VR users.

Although the existing works have made various efforts to build a useful volumetric video streaming system from different perspectives, there are several gaps between the existing research and a practical volumetric video streaming system that should

be both efficient and robust. First, the bandwidth efficiency issue: a volumetric video streaming system is expected to consider both network fluctuations and the viewer's viewing behaviors using an adaptive mechanism. Although the MPEG V-PCC provides compression codecs with good rate-distortion performance, there have not been any efforts to build a V-PCC-based volumetric video streaming system. This is partly because of the lack of a rate-distortion model that allows the MPEG V-PCC to adapt its compression parameters under different bitrate thresholds. Furthermore, even if a rate-distortion model is proposed, MPEG V-PCC is not suitable for providing a smooth playback experience due to its limited ability to scale the frame rate during the run-time. The complex three-stream dependency between the attribute, geometry, and occupancy maps further increases the challenges of implementing a passive frame-dropping mechanism that allows smooth playback of volumetric video without interrupting the user-watching process under network fluctuations. Another challenge of the volumetric video streaming system is its limited generalization ability. For a tile-based volumetric video streaming system, the prediction accuracy of the viewport and user behavior has an notable impact on system performance. Under a mixed reality setting, this assumption could easily be breached by changing one of the factors: network environment, viewing space, or the volumetric video sequence content. This has led to a drastic performance drop in the existing systems when tested under a new environment. In practice, building a prediction model that suits all possible deployment settings is infeasible. We notice the rise of few-shot and zero-shot learning algorithms in sequence modeling. Therefore, we can get a generalized system by reformulating the volumetric video streaming into a sequence prediction problem (FewVV). Finally, the robustness issue when transmitting the volumetric video over the unreliable network channels, as the latency requirements gets stringent for volumetric video, the partial reliable network channels provides a lower latency yet lossy transmission. To improve the robustness of volumetric video streaming process, an error concealing model could be helpful, however there lacks a dataset that support the effective training and benchmark of the error concealing models for volumetric

video. We build the first of its kind dataset to fill in this gap, and provide an in-depth analysis into the corruption patterns.

In this chapter, we first briefly demonstrate the problems of volumetric video streaming in Sec. 1.2. Then, we propose an overall research framework in Sec. 1.3, we summarize the contributions made in this thesis in Sec. 1.4, and finally, we organize this thesis in Sec. 1.5.

1.2 Research Problem

How to stream the Volumetric Video Effectively over the dynamic and limited network resources? The volumetric video represents an important type of immersive media. The fundamental challenge is balancing the high quality and high volume data across the limited network resources. A practical system requires an effective prediction of bandwidth dynamics, user viewport, and sequence salience. To achieve a smooth playback, it is also important for a volumetric video streaming system to have both coarse-grained and fine-grained adaptation ability. How to design such a system is challenging.

How to achieve better generalization? The existing volumetric video streaming systems rely heavily on the accuracy of viewport prediction accuracy. However, the real-world deployment environment is highly diversified, and it is thus a challenging problem to achieve a higher generalization level for the volumetric video streaming system.

How to transmit the volumetric video over lossy network channels that provides low latency? As the demand for higher interactiveness in volumetric video streaming increases, the latency requirement becomes more and more stringent. To this end, it is necessary to use unreliable (lossy) network channels, which do not

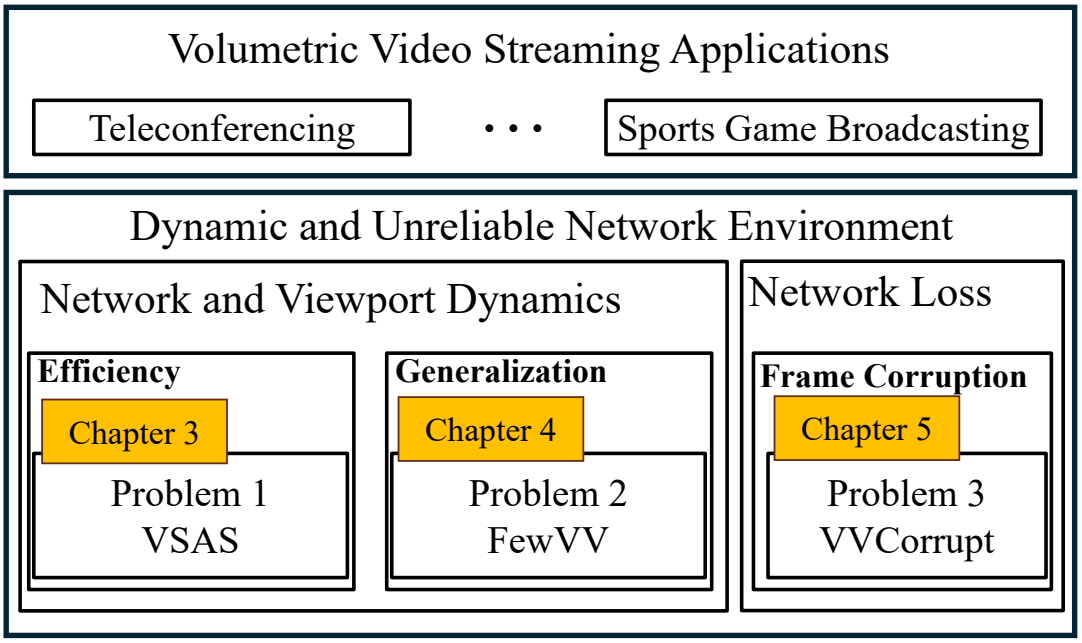


Figure 1.1: Research Framework

use a re-transmission mechanism and achieve a low latency. However, the downside of using these channels is their unreliability. The application layer decoders of volumetric video will be exposed to corrupted bitstreams, causing undesirable frame artifacts. Properly mitigating these errors' negative visual effects is a significant challenge for a robust volumetric video streaming system.

1.3 Research Framework

In this section, we demonstrate the overall research framework of this thesis. See Fig. 1.1, the overall research framework. The applications of volumetric video streaming systems, e.g., the teleconferencing, the sports game broadcasting, and remote support are deployed over the dynamic and unreliable network environments (the Internet) and viewed in the diversified virtual/augmented reality scenes. There are two types of adversarial challenges in this Internet, one is the bandwidth fluctuation

and scene diversity, and the other is the unreliable (lossy) network channels. In this thesis, we mainly focus on solving the bandwidth efficiency, scene generalization, and robustness over the unreliable networks.

To solve the first problem, bandwidth efficiency and fluctuation, we leverage the most recent development on the volumetric video compression standard MPEG V-PCC, which achieves high bandwidth efficiency but is not ready for video streaming services. We propose a **On-demand Volumetric Video Streaming with passive frame dropping (VSAS)** framework, where we design an on-demand video service framework with mixed coarse-grained adaptive bitrate control and fine-grained frame dropping to achieve a scalable, adaptive, and smooth volumetric video streaming system.

To resolve the second problem, generalization to unseen environments, we design a **Few-shot Adaptive Bitrate Volumetric Video Streaming with Prompted Online Adaptation (FewVV)** algorithm to generalize the volumetric video streaming. By reformulating the video streaming to a form suitable for transformer, we utilize the strong generalization ability of transformer models to achieve a generalized adaptive video streaming system.

Finally, for the third problem, we build a **A Bitstream-corrupted Volumetric Video Dataset for Partial Reliable Error Concealing (VVCorrupt)**, that includes a wide range of volumetric video sequences corrupted using our bitstream-level corruption model reflecting real-world network conditions. We made an in-depth discussion of our collected dataset to highlight the patterns of network corruptions on the decoded volumetric video artifacts.

1.4 Contribution

In this thesis, we study the volumetric video streaming and present innovative systems to address the challenges. To be specific, we made three major contributions:

1.4.1 MPEG V-PCC based bandwidth efficient volumetric video streaming

Our first contribution is proposing a bandwidth-efficient volumetric video streaming framework that, for the first time, allows DASH-based video streaming of V-PCC formatted volumetric video streaming, which is largely backward-compatible with the existing hardware acceleration chips and existing HTTP video streaming CDN infrastructure. We fill up several blanks. First, we develop one of the earliest rate-distortion model for V-PCC; second, we propose an offline reinforcement learning method to control the Bitrate according to the network dynamics; third, as the coarse-grained DASH architecture causes frequent frame freezing, we propose a DAG-based frame dropping mechanism that enables the existing system with a frame rate scaling capability. Together, our VSAS framework can deliver a smooth Internet volumetric video streaming service.

1.4.2 A generalized volumetric video streaming control with prompt-based tuning

Our second contribution is to study the generalization problem in tile-based volumetric video streaming systems. We first identified the limitation of the existing system when facing an out-of-distribution environment, which essentially constrains the real-world deployment of the tile pruning-based optimization of these systems. To tackle this challenge, we first reformulate the volumetric video streaming systems

control problem into a multi-variate sequence modeling problem, then train a causal transformer model with prompt-tuning to solve it. Our evaluation demonstrates a consistent improvement compared to several baselines regarding the QoE and the adaptation speed to an unseen environment.

1.4.3 A Bitstream-corrupted Volumetric Video Dataset for Partial Reliable Error Concealing

Our third contribution is to build a bitstream-corrupted volumetric video dataset for partial reliable error concealing. As the demand for more interactive volumetric video streaming services increases, the unreliable and partially reliable network channel is introduced to reduce the latency, which exposes the volumetric video decoder to possible corruption in the bitstream. We first propose a volumetric video corruption model that effectively reflects the patterns of transmission-induced corruption. Based on this model, we corrupt three types of volumetric video sequences with different contents and diversified spatial features, and then We made an in-depth discussion of of the visual effects of these corruption-induced artifacts. Finally, we outline practical guidelines for designing an ideal learning-based error-concealing model.

1.5 Chapter Organization

The organization of this thesis consists of following chapters:

Chapter 2 Background and Literature Review. We present the background and literature review of Volumetric video streaming in this Chapter, including the definition, existing architecture, and the primary challenges in this field, which include the performance issue due to limited network resources, the generalization problem due to an immersive viewing space, and the robustness issue due to a higher demand

on interactiveness. We present a comprehensive review of the literature addressing these three challenges. Which mainly covers the existing adaptive video streaming approaches and privacy protection schemes. We also discuss the problems of directly using these schemes on immersive applications and point out several preliminary solution approaches.

Chapter 3 VSAS. We focus on the performance issue of volumetric video streaming in this Chapter. As a new compression format, MPEG V-PCC is being standardized by the MPEG, and it has revealed the potential to become an important mainstream codec for volumetric video. However, due to its early development stage, it is not well supported and cannot easily fit into the existing HTTP video streaming infrastructures. Therefore, our work designs one of the earliest Internet volumetric video streaming systems for MPEG V-PCC format. We tackled three major issues: first, the lack of a rate-distortion model to facilitate rate control; second, the need for an offline reinforcement learning algorithm, which is suitable for network adaptation problem; third, to address the smoothness problem caused by the control granularity mismatch, we propose a DAG-based frame scaling method. Finally, we evaluate our system on various network conditions, video sequences, and viewer behavior datasets. The results illustrate a large gain in quality of experience (QoE).

Chapter 4 FewVV. We explore the generalization problem of the volumetric video streaming system in this chapter. We first measure the existing systems and identify the impact of the generalization problem in the viewport prediction of the volumetric video streaming. We reformulate the volumetric video streaming to be suitable for sequential predictor, and then, we solve it by proposing a volumetric causal transformer with prompt tuning. The evaluation shows a faster adaptation to the new environment, thus providing a higher generalization ability.

Chapter 5 VVCorupt. In this chapter, we studied the error-concealing problem of the volumetric video streaming system. We first identify the importance and necessity of error-concealing algorithms in interactive and low-latency volumetric video streaming systems. We then propose a novel bitstream corrupted volumetric video dataset to support the benchmarking and training of the error-concealing algorithm. We made an in-depth discussion of of our dataset's corruption patterns and point out the practical guidelines for designing a learning-based error-concealing model. Our dataset provides a solid foundation for developing learning-based error-concealing algorithms for volumetric video.

Chapter 6 Conclusion. In this Chapter, we conclude the thesis. We present some limitations and how well these techniques could be used in practical applications. We point out several future directions that could be improved in the next step of research, which could provide higher performance, better privacy protection, and more instant feedback between people in immersive video environments.

Chapter 2

Background and Literature Review

2.1 Volumetric Video Streaming

Volumetric video streaming is an important VR application. It pushed the 360-degree video's 3 Degree-of-Freedom experience further to 6 degree-of-freedom, allowing a viewing direction to be free and the user to move the origin point of view. Compared to similar techniques, volumetric video has a more accessible ground. It mainly relies on developing two technologies: the RGB-D depth camera and the VR headsets (Oculus Quest 2, Hololens 2). As these two devices become widely accessible to consumers, volumetric video streaming becomes a readily deployable service. Various immersive applications can be built upon the volumetric video streaming systems. For example, telepresence is a holographic teleconferencing application that brings geographically distributed people together in a shared virtual world. Telepresence has been proved to facilitate more effective emotional communication and connections between people, which could benefit education and professional collaborations. Sports Live Broadcasting is another example, where a sports game is recorded from multiple angles with high-resolution cameras, a background processing system transcoding it into a volumetric video, and then streamed over the Internet. Boxing and football

are two pioneer sports that introduce these experiences.

The research community and the industry have made a significant effort to standardize and improve volumetric video streaming. The foundations of any streaming system are codecs. There are two mainstream standards: First, the MPEG G-PCC standard, which organizes the point cloud using a tree-based data structure, allows effective incremental coding for the spatially sparse point clouds; however, this standard does not consider the temporal consistency in compression logic. Streaming volumetric video as independent frames using the G-PCC standard is still possible, but the compression efficiency is limited. Second, the MPEG V-PCC standard, that compress the point cloud as 2D video streams, e.g., a group of static point clouds arranged according to the timeline with constant intervals, each static point cloud is an analogy to the frame in the 2D video compression. This standard first projects the point cloud into several agnostic patches, and for each patch, it retains depth and color information with some metadata to allow reconstruction. This approach allows effective reuse of the existing 2D video codecs with very strong temporal compression ability, and it achieved an outstanding compression ratio that can stream the common quality volumetric video over the 4g/5g bandwidth. MPEG V-PCC also features better backward compatibility as there is a wide range of hardware acceleration for the underlying codecs.

Based on the two volumetric video coding standards above, the video streaming systems have been developed [12, 74] and optimized [63, 29] to adapt to the network and user-watching behavior dynamics. The decoding efficiency issue on GPUs has also been studied to reduce computation overhead [28]. The quality-of-experience model used in streaming optimization has also been a research focus [32, 34, 63], reflecting the need for different streaming architectures.

2.1.1 Traditional video streaming control

video streaming is a problem that finds an intersection between the dynamic network conditions (bandwidth, delay, and packet loss) and the video quality. Intrinsically, the 2D video streaming system should maximally utilize the available network resources to deliver a highest video quality, which is widely measured as quality-of-experience. However, this problem varies according to different scenarios with different application requirements. Broadly categorized to three types, the Video-on-demand services, the live-streaming services, and real-time services.

VoD Streaming Control Video-on-demand service is a service that playback a pre-stored video, like movie, from an content-distribution-network (CDN) server, it has a loose delay requirements, since the video is already stored, it is a single-way service. However, a high quality and a high frame rate is expected, as well as a smooth playback experience. Robust-MPC [70] first formulated this problem into a model predictive control process's. Then the rate-based [19, 58] and buffer-based [15, 56] methods are proposed later, further the unified methods using learning-based algorithms [6, 25] are proposed to use more advanced techniques. Recently the Meta-learning and Meta Reinforcement Learning are introduced to this community.

Live-streaming Control LiveNet [30] tries to design a low-latency video transport network that scales better than WebRTC-based systems while achieving lower latency than DASH-based systems that helps to meet the requirements of the massive live streaming with strong interactive needs. Tightrope [57], explored the playing speed problem on top of the existing rate control systems, which opens a new dimension for the quality-of-experience optimization. There are several other works on live streaming [30, 46]. Recently the meta-learning has matured and being introduced to the live-streaming communities, which is based on the personalization effect observed in several aspect of the video streaming systems, for example, MERINA [22] takes

a meta-learning approach to achieve an online reinforcement learning with strong generalization ability. MultiLive [64] consider a server-driven approach to support a large scale bitrate and loss control for the Live Video Streaming.

Real-time Video Control OnRL [78] propose the first reinforcement learning (online) system to facilitate real-time video telephony, with several designs to alleviate the exploration cost arising from the random sampling process. QARC [16] propose to control the sending rate only with deep reinforcement learning, and it also optimizes the quality of the video. Tambur [50] pushes this further to consider loss recovery directly inside the codec, which achieved much better performance by utilizing several codecs specific design. Loki [77] first identifies the long-tail distribution of the learning-based real-time communication(RTC) systems. It then proposes a mixture of rule-based methods to mitigate such an effect, outperforming the end-to-end reinforcement learning approaches. There are also several other works on congestion-based RTC [62, 80].

2.1.2 Volumetric video streaming control

Volumetric video streaming control is a more complex problem than traditional video streaming problems. At the current state-of-the-art level, on-demand video streaming is a mainstream service, while live-streaming and real-time streaming remain challenging applications to implement. There are three volumetric video streaming system approaches based on different compression formats: MPEG G-PCC (tile-based), MPEG V-PCC, and Mesh. We introduce existing works on these three approaches below.

Tile-based. Vivo [12] proposed three *visibility-aware optimization* to reduce the volumetric video Bitrate while minimizing the negative impact on QoE. It is the pio-

neering work on octree-based point cloud video streaming. GROOT [28], also based on octree format, modified the traditional octree data structure to support parallel encoding/decoding on mobile GPU, drastically improving the decoding speed. Lisha Wang et.al. [63] proposed a rolling window method with DRL. It instantiates a novel prediction-optimization-transmission (POT) framework inspired by the rolling window principle. This closely fits into the GoF structure of volumetric video streaming. They use a DQN-based deep reinforcement learning for each rolling window to control the Bitrate. Jie Li et.al. [29] proposed a QoE-driven adaptive streaming approach. It achieves two goals: first, it achieves a high QoE by modeling a perspective projection, and second, it reduces the transmission redundancy. Their primary contribution is a new and comprehensive QoE function considering several indicators, including spatial position, occlusion, and device resolution. Yu Liu et al. [34] proposed a practical mobile volumetric video streaming system using a multi-view transcoding approach. It differs from the Vivo because it does not directly decodes the point cloud on the client device, instead it adopts an edge server-assisted streaming paradigm. The point cloud is streamed from the cloud to the edge, the decoding and rendering are finished on the edge, and then the client is fed by a 2D video streaming according to its viewing coordinates. Their work pushes this architecture further by pre-fetching and computing a range of candidates' views instead of a single view, which allows a faster reaction given a new FoV of the viewer. They also propose a QoE model to facilitate this process. Anlan Zhang et.al. [74] in their work Yuzu choose a very different direction to improve the point cloud streaming quality. They use point cloud up-sampling and coloring to balance communication and computation. The system first down-samples and compresses the volumetric video into low resolution and low quality to meet the tight network bottleneck. Then after the receiver receives the low-quality point cloud, it up-samples the point cloud to a high quality. They solved the coloring problem of the existing works, and evaluated this idea using a prototype system.

V-PCC-based. MPEG V-PCC [39] is a new way of compression the dynamic point cloud. It first projects the point cloud into three planar images, the attribute map(color), geometry map(depth), and occupancy map(metadata), with a group of optimization processing, it uses 2D video codecs, e.g., H.264/AVC to compress the resulting image stream. When the receiver receives the stream, it reconstructs the 3D point cloud according to the projection metadata and the three-plan image stream. By doing this, V-PCC achieved a very high temporal compression efficiency. Yet, the existing support for V-PCC to work on the streaming is limited. Shuang et.al. [53] propose to use down-sampling and super-resolution to achieve a low bitrate V-PCC-based volumetric video streaming.

Mesh-based. Free viewport video [8] is a mesh-based volumetric video streaming system, which takes a surface reconstruction procedure first to translate the point cloud into triangle meshes and then register the texture on these triangle surfaces. In this work, a group of heuristic improvements is proposed to achieve a higher level of experience at a lower computation cost and network consumption cost. First, they propose an adaptive meshing scheme that localizes the region of interest (RoI) within a mesh, e.g., an actor's face, before giving the RoI a higher meshing density. Such a design allows a higher resolution and more detail for those salient areas within a volumetric video. Second, they design a temporal incremental coding scheme that uses key and incremental frames to achieve an effective temporal compression. Since the content of the volumetric video is often temporally consistent, e.g., a person dancing in a scene, the similarity between consecutive frames is very high. Therefore, storing each frame independently and with full information is unnecessary. Although the idea of this compressing scheme is the same as 2D videos, it carries a unique challenge: maintaining the meshing consistency between the frames, which is non-trivial for existing meshing algorithms. And finally, they propose an effective parallel computing system to process these data. Bitrate adaptation and bandwidth prediction are based

on a naive design that is sufficient for a meshed system.

2.2 Quality-of-Experience Metrics

It is important to have an optimization objective, which is the user's satisfaction. It is directly measured by the MOS (mean opinion score), it is typically collected using a survey, commonly at the end of a video play. However, since this metric is sparse and can hardly be connected to the objective parameters (resolution, frame rate, encoding rate, and latency), a model that maps these measurable metrics to the MOS is proposed, called the quality-of-experience (QoE) estimation model. We adopt these models as metrics across the evaluation of this thesis.

2.2.1 QoE Factor Balance

Vue [34] conducted a subjective study on the QoE of the mobile volumetric systems. They first organize a group of subjects (viewers) to watch the video sequences using their system running on the VR headsets. Then, they collect the objective metrics during this viewing process and form a dataset. Finally, they asked the subjects to answer a survey and collect the subjective MOS. By mining and analyzing this collected dataset, they point out several key factors that impact the volumetric QoE. That includes the Viewport Smoothness S_V, the Motion-to-Photon Latency L, Resolution R, Viewport Drift D_V, and Stalling B. The relative weights of these impacting factors are learned by regression in the collected dataset. QoE-DAS [63] takes a different path towards a QoE function to facilitate the tile pruning-based volumetric video streaming over MPEG G-PCC, the primary factor to consider is the user's viewing direction and viewing frustum parameters, because whether a tile is located inside the viewing frustum of a viewer directly determines the relative importance of this tile. Also, it is necessary to consider the occlusion problem and the perspective

projection caused by distance effects. QoE-DAS discovers that these challenges can be addressed by migrating the computer graphics rendering pipeline. Therefore, they proposed modeling the QoE using the perspective projection matrices and the occlusion detection approaches in Computer Graphics. In this way, they proposed the first analytical model of the QoE Factor Balance. CaV3 [32], on the other hand, tried to extend the traditional 2D Adaptive bitrate streaming (ABR) QoEs to the 3D scenarios to facilitate the caching acceleration of volumetric video, the relative importance between the tiles are of particular interest in this QoE function. The core idea is to separate the tiles into high-quality and low-quality tiles. For each set of tiles, the a VQA metric is used, then they give different weights to these two sets of tiles and conclude caching-oriented QoE functions. All these functions are widely dependent on video quality assessment metrics like PSNR to evaluate the relative quality of a single frame or tile.

2.2.2 Video Quality Assessment Metrics

There are two main paradigms to evaluate the relative quality of a streamed volumetric video frame. The first is the fully referenced video quality assessment metrics, which depend on comparing the original uncompressed and compressed video frames. By formulating different features from the contrast of the original and the processed frame, the fully referenced VQA metrics can accurately reflect the quality level of a video frame. In a practical video streaming system, an original uncompressed frame could be sent over the Internet as a sample for the quality and can then be used as an optimization factor. The second is the non-reference video quality assessment metrics, which do not need an original reference frame to predict the quality. These models are mostly learning-based and achieve a slightly worse performance than fully referenced models, but they also have the advantage of a wider application scenario.

Full-reference VQA: PSNR-I [27] proposed the PSNR for 3D geometry information,

which is a 3D extension of the traditional PSNR, which considers the geometry information difference with a unique function. PCQM [36], on the other hand, explored a learned balance between the geometry and texture features. They first propose a group of features that involve contrast, lightness, and the difference between the texture, as well as the local geometry features like curvature, curvature structure, and so on. Then, through organizing a subjective study, they find the relative importance of these features. They conclude that lightness comparison and lightness structure in texture reflect the most importance of texture, and curvature structure carries the highest importance of geometry. They give a group of empirical parameters for reference. Their evaluation of the correlation between PCQM and Mean Opinion Score shows that PCQM outperforms other fully referenced VQA metrics by a large margin. There are also point-to-plain PSNR and point-to-point PSNR [59] that are widely used in the standardization activities of MPEG Immersive Projects, which mainly uses the distance between the points and plane to give an error before adopting the log scale transformation of PSNR to get a final score, although this metric lacks the consideration on the color features, it is very stable and widely used when assessing the volumetric video compression codecs.

Non-reference VQA: IT-PCQA [68] belongs to the no-reference method, which uses domain adaptation to transfer the 2D non-reference method to the 3D domain. It uses an H-SCNN neural network architecture and a generator-discriminator design to train a meaningful VQA model, which does not rely on the reference frame.

Chapter 3

An On-demand Volumetric Video Streaming System with Video-based 3D Codec

3.1 Introduction

Immersive video, represented by 3D volumetric video, has recently emerged as a promising application in the Mixed-Reality industry, serving as the technology foundation of AR, VR and the future Metaverse. According to the market report [35], the volumetric video market value is now 1.5 billion and will reach 4.9 billion in 2026. The leading Internet giants have already set up their strategic plans in this field, for example, the Holopresense project [44] from Microsoft and Starline project [26] from Google.

Volumetric videos, like 2D videos, are comprised of a sequence of frames. However, different from 2D frames consisting of pixels, the content of the volumetric video frame is 3D, represented by 3D points, Meshes, or other 3D formats. It essentially is a real-time captured 3D model of a scene. Therefore, viewers can freely navigate the

video. Such navigation includes 6 DoF, including three translational DoF (up-down, left-right, forward-backward) and three rotational DoF (yaw, pitch, roll). A group of innovative applications can be delivered through volumetric video, e.g., remote teaching and immersive live concert. This 3D content is typically viewed using Head-mounted-display(HMD) devices, e.g., Oculus Quest II.

Given the tremendous volumetric video size, various compression solutions are proposed for affordable transmission over Internet, where MPEG V-PCC is a promising one. MPEG V-PCC, proposed by MPEG, is designed for commodity-quality volumetric video streaming. It achieves a high compression rate and fast decoding by facilitating the widely-deployed hardware acceleration chips for H.264/AVC and H.265/HEVC codecs. It adopts the 3D to 2D projection method, followed by a traditional 2D video encoding pipeline to maximally utilize these existing chips while achieving a state-of-art temporal prediction coding ratio. Generally, MPEG V-PCC is able to achieve an average of 30 to 90 times compression over ply format point cloud raw data.

Due to the increased immersiveness and interactivity, users' quality of experiences (QoEs) in volumetric video have changed drastically compared to a 2D scenario. One of the most important factors that affect QoE is the stalling effect. It is a common effect in 2D video-on-demand systems and happens when the network speed is lower than the video's intrinsic encoding rate. However, in 3D scenario, due to more strict delay requirement and more immersive watching experience, this effect will cause more significant QoE degradation. Thus, the traditional DASH-based chunk-level bitrate adaptation strategy, which works well in 2D video system, is no longer suitable in the 3D volumetric video service.

We argue that more *fine-grained* streaming configuration strategy is in urgent demand so as to satisfy the smooth QoE in 3D volumetric video watching. However, all existing works ignored this issue and mainly focused on other streaming optimization. For example, Vivo [12] proposed to improve Draco, an Octree-based direct

point cloud compression, with several viewport-based heuristics, to save bandwidth consumption. The free-viewport video [8] used mesh representation and proposed face detection-based heuristics to adaptively control the mesh density in order to better adjust the streaming decision. Such streaming adaptation works can only optimize network transmission from a macro perspective, while solving the network condition fluctuation problem within a chunk time remains a key challenge.

On the other hand, the *coarse-grained* bitrate adaptation of on-demand video streaming differs from 2D video streaming because of the difference in viewing devices. As users tend to use XR devices to watch the video, they are encouraged to move around during the watching. However, the new generation of Wi-Fi technology is sensitive to the occlusion and the relative position between the router and end device due to its bean-forming technology and higher frequency wireless signal usage, see Fig. 3.1. This makes it important to take the user's movement data into consideration in bitrate estimation. However, the complexity of movement slows down the convergence of any online reinforcement learning-based controller. We thus choose an offline reinforcement learning controller (decision transformer), which learns on the offline collected large dataset rapidly while fine-tuning online periodically with high data efficiency.

In this paper, we, for the first time, try to address the adaptive video streaming issue in MPEG V-PCC. *To our knowledge, our work is the first DASH-based video streaming framework supporting MPEG V-PCC.* We will show that by combining the existing chunk-level ABR with a passive frame-dropping add-on, the DASH-based MPEG V-PCC can eliminate most of the stalling event while keeping the scalability of the DASH architecture. During this process, we faced major challenges. (1) there is a lack of a rate-distortion model at the 3D level. More specifically, this model establishes the relations between the 3D QoE, Bitrate, and the configurations of VPCC codecs, and (2) there lacks an understanding of the internal bitstream structure and the complex frame dependency within a MPEG V-PCC chunk, making it difficult to drop the frames wisely.

Our paper designs an Internet Volumetric Video Streaming Framework for a new format, MPEG V-PCC, which the MPEG has recently standardized. This format has a very high compression ratio and achieves high quality in terms of rate-distortion curve. However, due to its comparatively complex and coupled design, there is not yet any framework to allow an Internet Streaming of Volumetric Video in this format. To address this issue, we design a new rate-distortion model that fills the gap and allows MPEG V-PCC to fit in the DASH framework.

To mitigate the DASH's large chunk size-led stalling effect, we introduce a passive frame-dropping (skipping) mechanism. Although there is similar work on 2D DASH, MPEG V-PCC differs from it as it maps the 3D point cloud into three synchronized 2D image streams with complex dependency; therefore, we first model this dependency into a DAG(directed acyclic graph), then propose a method to drop the frame according to this DAG. This design removed most of the stalling, and since stalling is especially sensitive in HMdevices, this improvement helps to increase the QoE for a large margin.

The complex and heterogeneous environment of the VR headset ask for a more advanced Bitrate Selection method (ABR). We propose to use a language model inspired Meta Offline Reinforcement Learning algorithm Decision Transformer to solve the Volumetric ABR problem, we motivate and introduce the movement-awareness to the bandwidth predictor, use a fast environment-shift adaptation method, and build the first offline reinforcement learning trajectory dataset for V-PCC based volumetric video streaming system. We trained our model and evaluated it on a large corpus of volumetric video sequences, network environment, and the movement trajectory. The results show a significant gain.

We choose ChatGPT2-based DRL as it belongs to the category of the Meta Offline Reinforcement Learning and has a state-of-the-art performance. Compared to the online RL, Offline RL has two advantages, first the safety, the HMDdevice and Mixed Reality Environment makes the system works on human beings, sometimes when the

online DRL agent makes a drastically bad decision, it could harm the user physically, therefore it is favored that any DRL controller should not explore a random action online. In this case, the offline RL achieves a better safety, as it is only trained on a closed offline trajectory dataset (a group of running trace collected from an expert controller). Therefore, it does not need to explore a random action during the test time. Second, the training and adaptation efficiency, offline RL is able to consume a large data corpus and trained very fast, while online DRL would have to learn by a lot of random explorations on real system with human, this is time consuming as each feedback could takes a long time. (the simulator-based online DRL training, however, cannot capture the true systems' behavior and thus achieves a sub-optimal result). On the other hand, thanks to the inherited generalization (Multi-task) ability of the GPT2, our proposed method is able to adapt to a new environment faster than the existing Meta-learning based algorithms.

We conducted an in-depth study on the MPEG V-PCC. We built two models to tackle these challenges: (1) a new volumetric Rate-Distortion Model that depicts the relationship between the encoding parameters, mainly quantization parameter (QP) and down-sampling rate (Dr) with the resulting files Bitrate (Rate) and PSNR (Distortion). So that given a Bitrate target, we can choose the optimal encoding parameters that maximize the PSNR of a single MPEG V-PCC video chunk. We systematically validated our Rate-Distortion model on a famous volumetric video dataset (8i Voxelized Human Body) with eight sequences. The results show high accuracy. (2) a new simplified graph model that depicts the frame dependency relationship between the three streams of MPEG V-PCC, e.g., the Attribute Map Stream, the Geometry Map Stream, and the Occupancy map streaming. We then design a Multi-constrained optimal path (MCOP) algorithm on this graph to compute the optimal frame order offline. We open-source these two models for future research in this field. Finally, we design a new volumetric video streaming framework based on DASH and using MPEG V-PCC as the codecs.

We evaluated our new VSAS framework on a large corpus of network bitrate datasets with a wide range of volumetric video sequences. We compared our framework with two state-of-art systems, the Vivo, an octree-based compression codecs supported system, and FVV, a mesh-based streaming system. We show that our framework outperforms these two baselines with a large margin of 1.67x on average. We **open-sourced** our code and dataset on Github *https://github.com/VSASproject/vsas*.

Our contributions are:

- For the first time, we tailor the existing 2D MPEG DASH video streaming to support the MPEG V-PCC-based volumetric video streaming in 3D scenes. We design a new Media Presentation Description format and a new Decision Transformer-based ABR algorithm to control the bitrate adaptation process.

- We introduce a new Rate-distortion model for MPEG V-PCC-based volumetric videos. We systematically validate our model on a large dataset and propose an optimizer that outputs an optimal encoding parameter combination given a bitrate target to satisfy the need of dynamic bandwidth.

- For frame-level adaptation, we design a graph-based model that can reflect the complex frame dependency in MPEG-VPCC. We use it to passively drop the unimportant frames when network capacity is insufficient due to in-chunk fluctuation.

- We implemented a VSAS prototype on Quic-go [7] and MPEG V-PCC reference software [39], we conducted a trace-driven evaluation of VSAS over a large-scale network traces corpus. The result shows a significant improvement of 1.67x compared to the three state-of-the-art volumetric video streaming systems, Vivo [12], FVV-Mesh [8], VPCC-Raw [10].

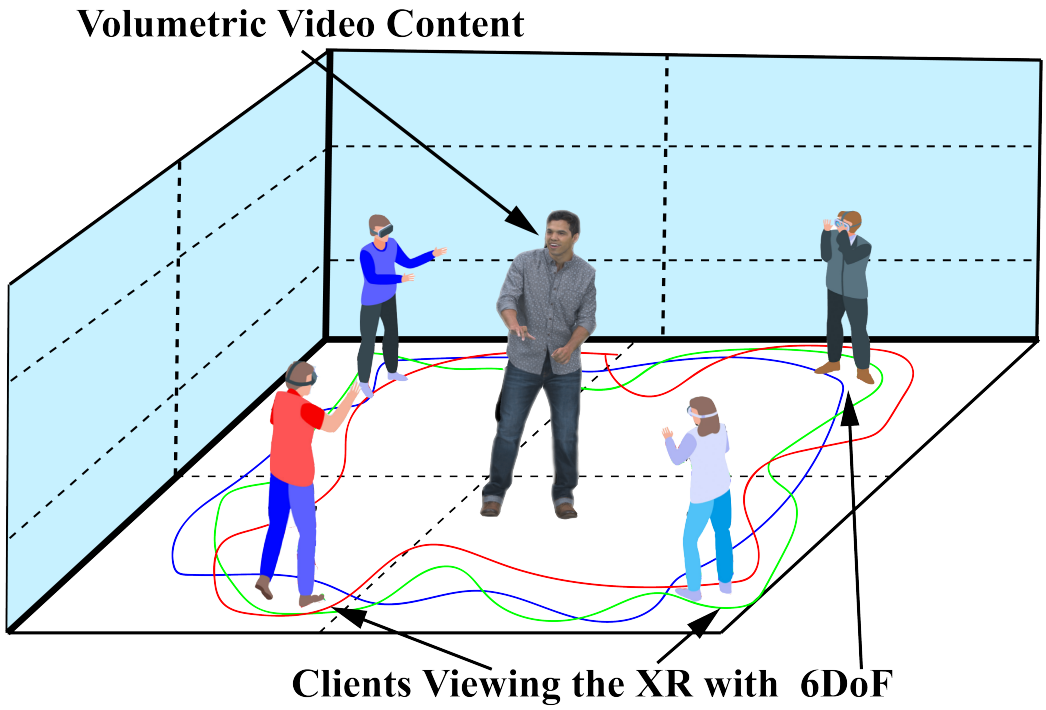


Figure 3.1: The environment occlusion causing Bitrate fluctuation.

3.2 Related Work

3.2.1 Background

Volumetric Video Streaming Vivo [12] proposed three *visibility-aware optimization* to reduce the volumetric video Bitrate while minimizing the negative impact on QoE. It is the pioneering work on octree-based point cloud video streaming. Our work differs from it as we use a more recent MPEG V-PCC point cloud format instead of the old octree-based Draco. GROOT [28], also based on octree format, modified the traditional octree data structure to support parallel encoding/decoding on mobile GPU, drastically improving the decoding speed, which is agnostic to our work. Microsoft High-fidelity Free viewport video [8] is a mesh-based volumetric video streaming system, which takes a surface reconstruction procedure first to translate the point cloud into triangle meshes, and then register the texture on these triangle surfaces. It is

not backward compatible with the existing 2D video streaming pipeline.

Decision Transformer As the GPT and BERT show strong performance on long sequence prediction in the language field, offline reinforcement learning, a type of sequential prediction problem, has evolved to use this strong sequence-to-sequence prediction capability. Given a target reward, it predicts the best action with the highest probability of reaching it at this step. In this way, a causal transformer(GPT) [5] can be used to memorize all the expert demonstrations that ever reached the target reward. It can find the most similar expert trace and outputs the action taken at this step by that expert demonstration. Repeating this process in an auto-regressive way will yield a online control process. DT encodes each token into an embedding and adds a positional encoding to each embedding. These aligned sequences of embedding are then fed into the GPT-2[49] Causal Transformer (it's possible to use GPT-3 and newer version of GPT here, but due to the speed issue, DT uses GPT-2 at present). GPT-2 applies an attention mechanism to predict a left-shifted version of the input: $(\langle \hat{s}_{k-W}, \hat{a}_{k-W}, \hat{g}_{k-W} \rangle, \ldots, \langle \hat{s}_k, \hat{a}_k, \hat{g}_k \rangle)$. This shifting is a key design component of the Decision Transformer. All these elements are available during the training on an offline collected trajectory of past replay. Such an offline trajectory can be collected from expert demonstrations or random trajectories. [1]

Volumetric QoE Vue [34] built a subjective QoE prediction model for volumetric video on a VR headset. Which systematically studied the factor that impacts the quality of experience of a volumetric video viewer using a Head-mounted-display (HMD) device. They considered Viewport Smoothness S_V, the Motion-to-Photon Latency L, Resolution R, Viewport Drift D_V, and Stalling B in their study and concluded the relative weight. Because The HMD device performance solely determines viewport-related metrics, we only adopt the relative weight of Resolution and Stalling in their study in our QoE model. We considered their viewport insights and stalling time-sensitively in our design. While PSNR-I [27] proposed the PSNR for 3D

[1] During inference, a_t is unknown and will be predicted in an auto-regressive manner.

geometry information, which we used in our evaluation.

Rate-Distortion Optimization Rate-Distortion Optimization and Rate-Distortion Modeling are mature research fields in signal processing and video coding community. It lies in the center of a trade-off between the encoded video rate and its quality(roughly defined as the inverse of distortion). The traditional works focus on the PSNR. For example, Singhadia et.al.[55]'s work established the relationship between the QP, Encoding Rate, and the Reconstruction PSNR for H.265/HEVC codecs and proposed a logistic model, and the result shows good accuracy. Our work differs from theirs, as we focus on point cloud video instead of 2D video. There is also 3D rate-distortion optimization. In their work, Xiong et.al [65] propose to use the occupancy map in MPEG V-PCC to guide the Rate-distortion optimization process, which results in a fast CU mode selection algorithm. Our work, however, chooses a direct modeling approach that targets the raw point cloud input instead of the projected 2D map.

3.2.2 Limitations of the existing system

DRL-based 360-degree video ABR systems. DRL360 is a pioneer work on optimizing the 360-degree video streaming with an in-depth study into the viewport, tiling and bitrate adaptation mechanisms with strong theoretical analysis. The Deep Reinforcement Learning was used to optimize the final adaptation. This paper studied a similar, yet different type of video media, volumetric video, which is a group of point cloud with a dynamic movement, that is rendered again to show. Both volumetric video streaming and 360-degree steaming provides an immersive experience on HMD devices. While compared to 360-degree video's 3 DoF(degree of freedom) experience, volumetric video advances with another three DoF.

However, they share lots of similarities, they both depend on the viewport prediction, and they both have a tile-based ABR mechanism on DASH. What's different

is that Volumetric video has a higher bitrate in its raw format, thus demanding a more efficient compression that is very different from H.264, and Volumetric video has movement in the viewport trajectory, makes it more challenging to predict the viewport for volumetric video.

Directly applying the DASH-based on-demand video streaming framework on volumetric video is not ideal. The main reason is the impact of the stalling effect. See Fig. 3.2.

The stalling effect happens when the buffer is empty, this effect happens frequently when the ABR's bitrate prediction is inaccurate, and it becomes more serious when then buffer is small. Which is common in the Live streaming applications, where the buffer is set to one to two chunks for a lower end to end delay.

On this occasion, the playback will freeze. This effect is ok in a 2D scenario because the user watches the video on TV. However, in a 3D volumetric scenario, it is unacceptable. Because users use an immersive VR headset to view it, such stalling can cause physical injury to the user. However, any DASH-based method can hardly remove the stalling effect because the DASH chunk makes the shortest decision interval of 5 seconds. We thus introduce a passive frame-dropping method to mitigate this stalling problem. As passive frame dropping will allow the chunk to be partially transmitted, and thus by cutting the tail frames can guarantee low stalling in any network circumstances. But MPEG-VPCC uses three different video streams with complex inter-dependency. We solve this in our design by building a new graph model. While we also adopted a more advanced ABR to improve the chunk-level bitrate decision accuracy.

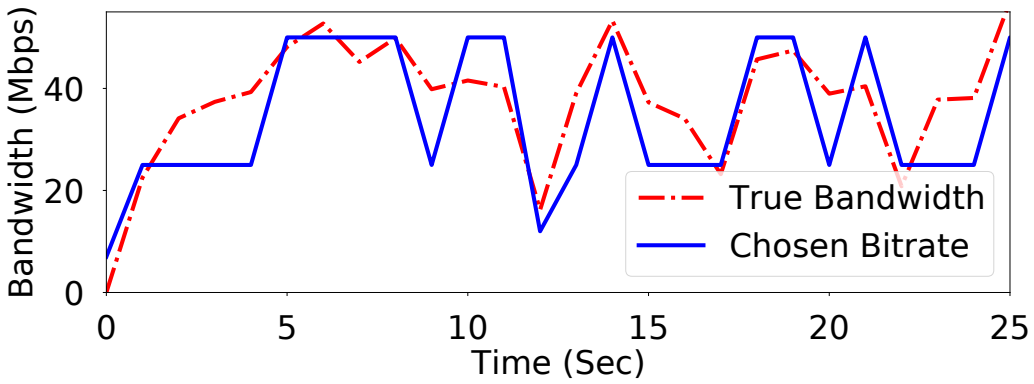


Figure 3.2: Root cause of stalling in One-step DASH.

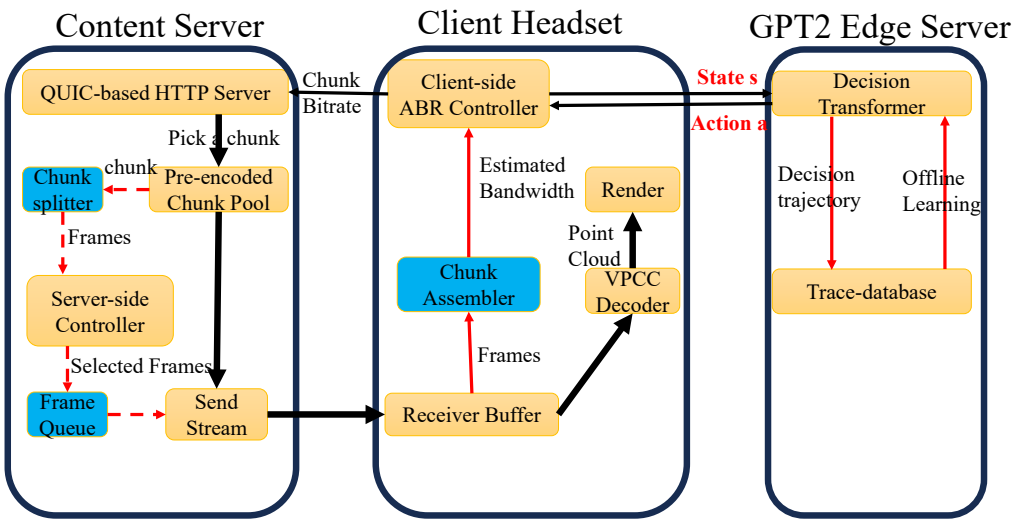


Figure 3.3: Volumetric ABR architecture.

3.3 Design

3.3.1 Design Overview

Fig. 3.3 illustrates the system framework. Which mainly involves two sides, the server and client sides. Note that the server side can scale to support multiple clients. This design is inherited from a standard HTTP service. While The client-side is a Unity script written in C Sharp, running on a HoloLens2. This design can easily extend to WebXR-based Android phones and other VR devices. As our framework is platform-independent. The client side is responsible for the runtime adaptation, which includes two modules. The first is the Bitrate adaptation module, which works on a chunk granularity and makes the per-chunk decision of point cloud chunk representation. The second one is a Frame Adaptation module responsible for the frame adaptation. This module is an add-on to the Bitrate adaptation module, as the Bitrate adaptation module works on a chunk granularity that equals a playtime of 5 seconds. If the network environment experiences a significant change during this 5 seconds interval, Bitrate adaptation cannot react. This is the root cause of the stalling effect. However, because Volumetric video is played on a VR headset, such stalling is minimally acceptable. So We introduce a Frame Adaptation module to handle this scenario. On the server side, the volumetric video needs to be compressed by the server into several MPDs, with different representation levels. We design a new representation here, see 3.4, which reflects three factors, the Downsampling rate, the attribute QP, and the geometry QP. However, there is no existing method to determine the combination of these parameters to achieve a target bitrate, so we introduce a new Rate-distortion model to facilitate this process.

Bitrate Adaptation module The bitrate adaptation module determines the bitrates of the playback process in a chunk granularity according to the bandwidth estimation and the viewport information. We choose a learning-based approach, which

integrates bandwidth estimation and decision-making into a single GPT model. Our analysis observes that the recently proposed offline reinforcement learning is a suitable class of algorithms for volumetric video streaming control. See the details in Sec.3.3.3.

Frame Adaptation module The frame adaptation module is a passive module that cuts the tail of the chunk when the actual network bandwidth runs lower than the estimated available bandwidth, which will cause a stalling without a frame adaptation module. This mechanism is beneficial for a weak network environment, where the bitrate adaptation module rarely gets a chance to accumulate enough buffer.

Server-side DASH Representation Format We designed a new DASH Media Presentation Description format for the MPEG V-PCC-based volumetric video streaming process. Which allows variable chunk size and frame rate. See Fig. 3.4. The dynamic point cloud sequence is captured from the RGB-D cameras or comes from a stereo-calibrated camera. Typically, such a capture system requires three or more cameras to capture a whole human body point cloud in real time. The resulting format is a point cloud. That is, each frame of the volumetric video stream is a static colored point cloud. The V-PCC standard first projects the point clouds to three planar images, including color and depth information from different projection angles and aliased onto a plane.

We first cut the video sequences into 5-second chunks with an original frame rate of 15-30FPS. This is equivalent to 75-150 frames per chunk. Then for each chunk, We compress it into N different representations(bitrate levels). For each representation, it has a target bitrate. By using the Rate-Distortion Optimizer proposed in Section.3.4, we find a set of encoder parameters that achieves this target bitrate. We then use a MPEG V-PCC encoder with FFMPEG support to encode the chunks to N different representations.

Finally, we generate an XML file called Manifest, including the URI link to each chunk

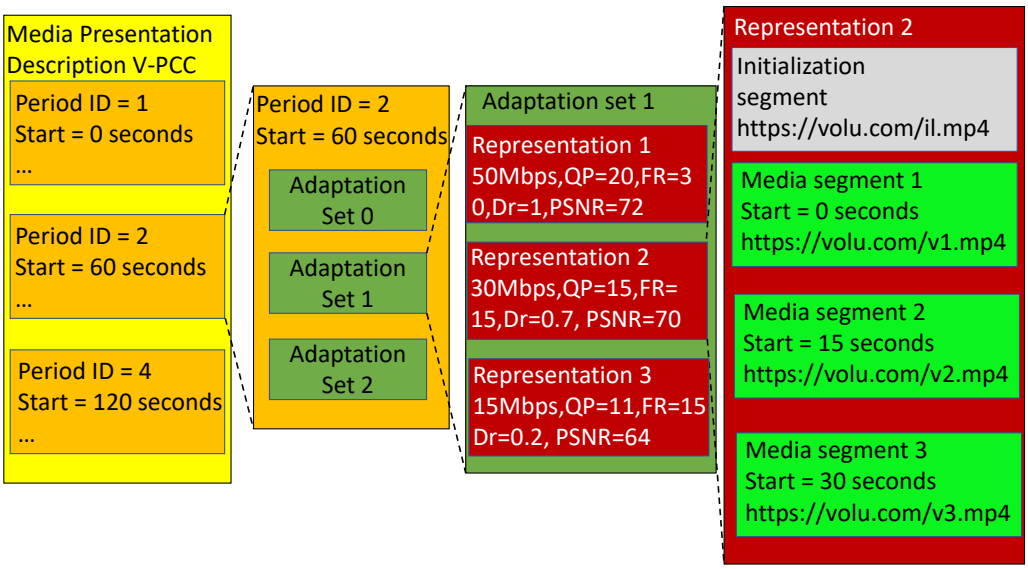


Figure 3.4: The Media Presentation Description Format for VSAS.

and a list of available representations. This design ensemble the design of DASH and HLS. An example Manifest and a Typical VPCC MPD is shown in Fig. 3.4.

3.3.2 QoE Model

QoE is the reflection of the humans' perception and experience over a multimedia content. In the scenario of the volumetric video streaming. It mainly includes three factors. First, the individual quality of the frame Q_i^0, which is the quality of a chunk c_i, this factor is directly related to the bitrate of the video, while, in this work, we use point to plain (p2plane)-PSNR as our primary metrics, that is a reference-based video quality assessment metric. It considers similarity between the origin content and decoded point cloud. The formula is listed as below:

$$Q^0 = \text{PSNR} = 20log_{10}(MAX_c) - 10log_{10}\left(E(f_i - \hat{f}_i)\right). \tag{3.1}$$

where f_i is the pixel value after compression, and $\hat{f}_i$ is the pixel value before the compression. So it is a log scale measure of the mean difference before and after a

compression/transmission process. For volumetric video, the basic unit is the point instead of pixel, so f_i is the point's value.

The second factor is the quality switching penalty, this factor measures the fluctuation of the video quality between differ net chunks. Human beings prefer a smooth and consistent watching experience, therefore, it is better to avoid unnecessary fluctuation of the quality shift. It is formulated as the change of the Q^0.

$$Q_i^1 = |Q_i^0 - Q_{i-1}^0| \tag{3.2}$$

And the third factor is the stalling time, which is the time that the frame frozen, which happens when the buffer is drained out. We defined is as $Q_i^2 = \max\{C_t - \frac{S_t}{B_t}, 0\}$. Where B_t is the available bandwidth and S_t is the bitrate. And finally the overall QoE is the linear combination of these three factors.

$$Q_i = \alpha Q_i^0 - \beta Q_i^1 - \gamma Q_i^2 \tag{3.3}$$

3.3.3 Bitrate Adaptation

Bitrate adaptation of an on-demand volumetric video streaming system differs from traditional 2D video streaming systems due to the *movement-caused network variation* problem and a higher sensitivity to the *stalling effect*. In 2D video streaming, the viewers tend to stay static (sitting on a chair watching a TV) during the viewing process, however in 3D, because of the nature of 6 Degree-of-Freedom navigation, the user will wear a VR headset and move around the viewing environment(for example, a classroom), see Fig. 3.1. Further, when multiple users share a viewing environment, their movement complicates the situation. On the other hand, Wi-Fi 6 and newer generations of wireless networks root heavily on high-frequency, beam-forming technology, regardless of its higher throughput. This technical decision has made this generation of wireless networks susceptible to occlusion because as the wavelength of a wireless signal gets shorter, its ability to bypass the obstacles weakens. To this

end, the user's movement will be essential when predicting the bandwidth between the Wi-Fi access point and the end device. [76]

Problem Formulation. Due to the similar binary file format and compatible codecs format, on-demand volumetric video streaming over MPEG-VPCC is similar to a typical ABR system. The video is cut into K chunks (i.e., segments), L represents the playtime of each chunk. Each chunk can be encoded with different bitrate $\mathcal{A} = a_1, a_2, \ldots, a_M$. For each chunk U_k, the bitrate assigned is denoted as $a_k \in \mathcal{A}$. Note that in each time slot t, the receiver side can determine to initiate a pull request to download the k-th chunk and put it to the receiver-side playback buffer $\mathcal{B}$, after U_k has been downloaded, the the buffer space (measured by residual playing time) of $\mathcal{B}$ is denoted as B_k. When $B_K < 0$, it means the player does not have the content to play at the moment, and it will lead to a *stalling event*. Stalling will cause the picture of the volumetric video to freeze and potentially hurt the user from either a physical or experience perspective. As can be seen in the above formulation, the key to solving this temporal decision problem requires an accurate estimation or prediction of average bandwidth during the next time slot C_k, and considering the current buffer occupancy B_k, to choose the best version of k-th chunk a_k, so that there won't be any stalling event, and the quality of play E_{a_k} is maximized.

This process is in fact a Markov Process. Where the state, s_k represented as the combination of the above mentioned features, and action being the representation of the k-th chunk a_k. and the reward $r(s_k, a_k)$ quantified as the linear combination of the stalling time and quality.

$$r(s_k, a_k) = q(a_k) - \beta(d_k - B_{k-1}) \tag{3.4}$$

Here $q(a_k)$ is pre-computed by using the p2plane-PSNR[18], and we have R-D model in Sec.3.4 to find the trade off between p2plane-PNSR and the bitrate E_{a_k}. It is now a reinforcement learning, which optimizes the sum of the quality r_k, subject to the

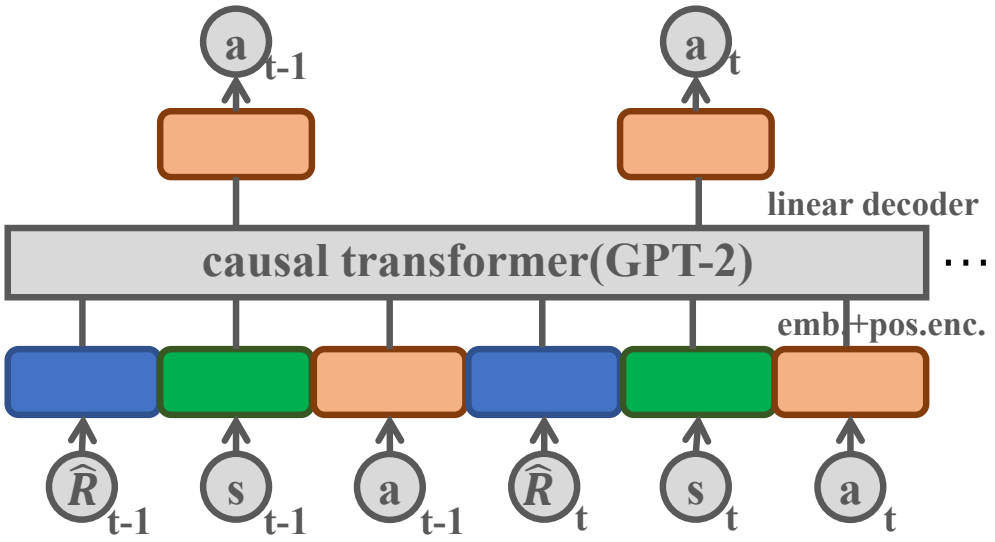


Figure 3.5: Decision Transformer

bandwidth constraint.

$$a_k^* = \arg\max_a \frac{1}{K} \sum_{k=1}^{K} r(s_k, a_k)$$
$$\text{s.t.} s_{k+1} = f(s_k, a_k), a_k \in \mathcal{A} \tag{3.5}$$

Where $f(s, a)$ is the state transition function that maps the state and action of the current time slot k to the states of the next time slot $k+1$. Note that, the state transition probability $0 \leq P(s_{t+1}|s_t, a_t) \leq 1$ is implicitly represented in f. Unfortunately, the analytical form of this function is hard to obtain because it involves the complex and heterogeneous network bandwidth variation and the personalized change of the user viewing position. Thus, we seek a learning-based model to learn and represent this hidden function parameterized by θ.

Decision Transformer for Volumetric ABR

Online and Offline Reinforcement Learning Due to its two traits, we consider offline RL as our system paradigm. First Offline Reinforcement Learning algorithm is more safe, as the volumetric video streaming is immersive, a wrong action of the ABR

controller could frighten the viewer and potentially cause physical injury to the user. Therefore, an online reinforcement learning's random exploration behavior should be avoided. Second, offline reinforcement learning allows a faster adaptation to the new scenario, because offline reinforcement learning has a faster training speed, a large trajectory dataset can be used, which allows the offline reinforcement learning to be more general and has a multi-task ability.

Now, we formalize the reinforcement learning problem. the RL problem is a decision making process, that aims to optimize the accumulated reward by finding a good policy. Each of this RL task can be modeled as a Partial Observable Markov Decision Process (POMDP). Let $M = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \mu)$. Where $\mathcal{S}$ is the state space, and each state $s \in \mathcal{S}$ is within this set, it is patially observable for the volumetric video streaming problem, as the underlying network status can only be estimated not truly measured. While $\mathcal{A}$ is the action space, in the volumetric video streaming task it is defined as the bitrate and frame rate of the next chunk. And $\mathcal{P}$ is the state transition matrix.

Under a online setting(the traditional setting), the RL agent is allowed to explore the random actions. The state would change accordingly, and send a feedback including the new states s_{t+1} and the reward r_{t+1}, and using a policy or value gradient method, the DRL agent could find a better policy. However, this process is time consuming, in Volumetric video streaming setting, each step could takes 2 seconds to finish (2 seconds per chunk). While a typical PPO(policy proximal optimization) agent could takes 100k steps to converge, this translates to a very slow training. When it comes to the simulator-based online training, simulator is not a true system, and could hardly imitate the behavior of a true system, thus any agent trained on simulator is constrained by the fidelity of the simulator. This approach is thus not ideal.

While for offline reinforcement learning, the system could first run on any policy defined. and then a trajectory $\{s_0, a_0, r_0, \ldots, s_T, a_T, r_T\}$ could be collected and form a dataset. Using only this dataset $\mathcal{D}$, making this problem more challenging than the

traditional RL problem.

Since the multi-task nature of the volumetric video streaming, as different people and different viewing environment would lead to a fundamentally different state transition metrics $\mathcal{P}$. For a set of tasks $\mathcal{T}$. $T_i \in \mathcal{T} = \{M_i, \pi_i\}$. That is each task T_i is corresponding to a partial observable Markov decision process M_i and a policy π_i. We use a double loop to minimize the in-task loss and the multi-task difference. See Alg. 1.

Difference between the online and offline reinforcement learning for Volumetric ABR problem. Online reinforcement learning is a learning algorithm that interacts with the environment and learns from the reward feedback signals. It gradually learns the optimal actions by exploring different actions under the given states. By balancing the exploration and exploitation, it could also avoid the negative impact of this learning process. On the other hand, offline reinforcement learning does not need to interact with the environment and relies solely on the previous offline trajectory dataset to find an optimal solution. Offline reinforcement learning is more challenging than online, as it needs to infer the full state transition by giving partial observations without access to the new samples. However, because of its higher sampling efficiency and safe behavior, it is widely used in autonomous driving and robotics by imitating the human expert.

Advantages and limitations of offline reinforcement learning. The main advantages of offline reinforcement learning are twofold: first, sampling efficiency. Because the solution space of the volumetric ABR problem is much larger than the traditional ABR problem, the sampling efficiency becomes a central problem in building a practical system. Offline reinforcement learning could train offline, compared to online reinforcement learning training on the simulator, it achieves faster training on a large solution space; second, safety, because offline reinforcement learning does not explore random new actions during the run-time, its behavior is more predictable, and therefore achieves a safer performance, which is importance on Volumetric ABR

system because Volumetric video is widely viewed on HMD devices, this means a poor streaming quality could potentially lead to physical injury. On the other hand, the primary limitation of offline reinforcement learning is its generalization ability, because the model is trained offline, and only do inference online, it has a limited ability to adapt to new environment.

The movement-awareness for bitrate prediction. Here, we explore the movement-awareness, see Fig. 3.6, the first row is the distance between the router and the headset, the second one is the angle, while third row is the corresponding bandwidth, we can see that the bandwidth varies according to both the distance and the angle between the router and the headset. Fig. 3.8 and Fig. 3.7 shows similar result on classroom and outdoor scenes. This motivate us to involve the movement and position as the input to the decision transformer, see Fig.3.9, the QoE can benefit greatly by introducing the movement features into the state of the decision transformer.

States. The state of the rate adaptation module at time slot k includes several factors. The first bandwidth estimation is denoted as B_k. Second, the playback buffer occupancy is denoted as C_k. Third, the playback deadline L_k. Fourth, the viewer's movement trajectory in the most recent W frames $\vec{V_k} = [V_k, V_{k-1}, \ldots, V_{k-W+1}]$.

Actions. The decision transformer will decide three actions for each chunk, first the quantitative parameters $\vec{\mathrm{QP}}_k$ for each block. Second, the point cloud density $\vec{\mathrm{Dr}}_k$ of each block corresponds to the video resolution in 2D. Third, the frame rate of this chunk M_k. For notation simplicity, we let $a_k = \vec{\mathrm{QP}}_k, \vec{\mathrm{Dr}}_k, M_k$ Note that these factors will determine the bitrate and quality of the volumetric video and will result in a reward, our newly designed MPD format(See Fig. 3.4) allows indexing using QP and DR as index. The bitrate is a function of these three parameters $f_{BR}(\vec{\mathrm{QP}}_k, \vec{\mathrm{Dr}}_k) \times M_k$, and the quality of the resulting chunk is measured by the receiver by using p2plane-PSNR score, a full-reference metric, it can be directly referenced from the MPD metadata.

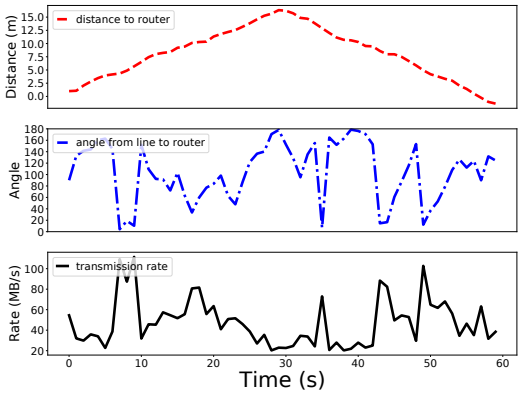


Figure 3.6: The relation between movement and bandwidth, home

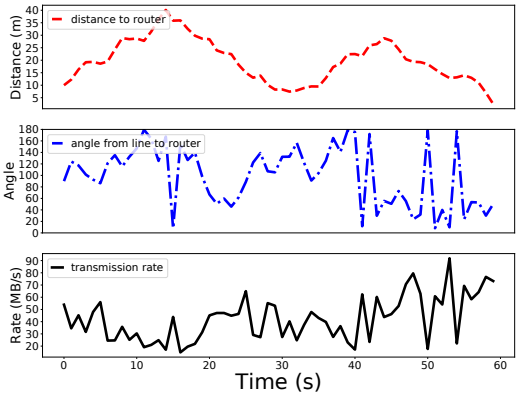


Figure 3.7: The relation between movement and bandwidth, outdoor

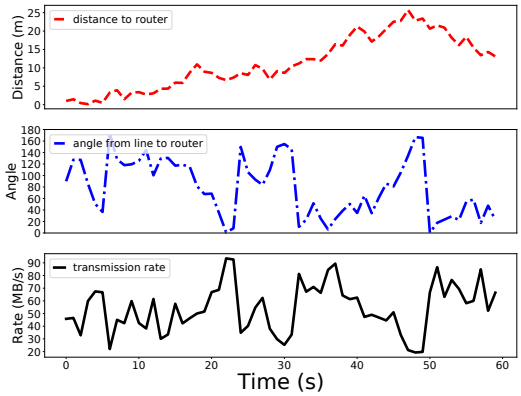


Figure 3.8: The relation between movement and bandwidth, classroom

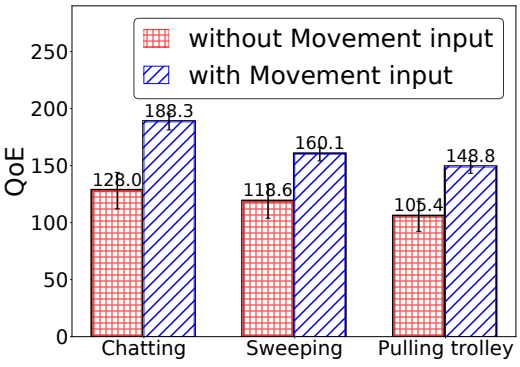


Figure 3.9: The comparison between movement-aware and movement agnostic controller

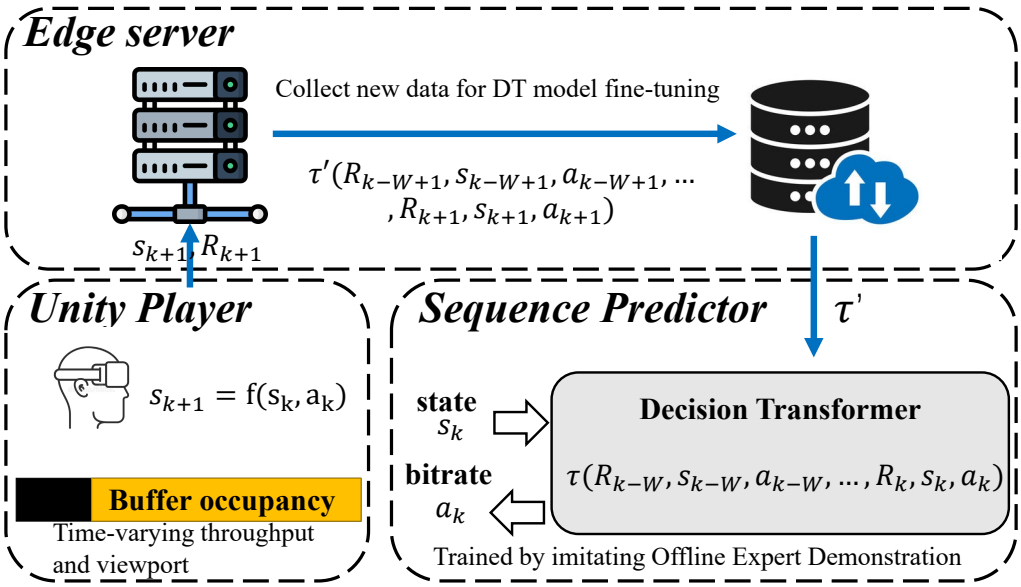


Figure 3.10: System Arch of Decision Transformer Controller.

Reward-to-go function. We use the reward function $r(s_k, a_k) = q(a_k) - \beta(d_k - B_{k-1})$, and let the expected reward be G^*, then the reward to go is computed as $\hat{r} = \sum_{i=k}^{K} r_i$ during the training, as we already know the reward of the future playback in the trajectory. During the inference, it is computed as expected reward deducts all past reward $\hat{r_k} = G^* - \sum_{i=0}^{k-1} r_i$.

Offline Dataset Collection and Pre-Training See Fig. 3.10, the collection and training architecture. The decision transformer is used to control the Unity-based VR player, while the VR player will collect the history trajectory and reward and update the offline training dataset on the edge server. The edge server will re-train the decision transformer with an updated dataset periodically. [2] The training algorithm is shown in Alg. 1. Here the loss of the Decision transformer is defined as

$$\mathcal{L}_{DT} = -a_k \log(\hat{a}_k[a_k]), \hat{a}_k \in \mathcal{A}$$

Where $\hat{a}_k$ is the DT predicted action at k-th chunk, a_k is the true-optimal action. The

[2]We note that there is a more efficient decision transformer that does not require a full model fine-tuning to adapt to the new environment. We leave this part to our future study[66].

decision transformer needs to be pre-trained before operation in real-world systems. We used two methods to generate an offline trajectory dataset for the pre-training.

Random. Here, as the decision transformer is able to learn from the random examples, we start our trajectories with a random token set. We first choose a network trace for each time slot k, and we get the bandwidth, and latency, then generate a grid of viewports, in a circle, with an interval of 5 degrees. Now for each state, we randomly sample N encoding parameters and M down-sampling rates from the possible datasets. We then compute the PSNR using the Rate-Distortion Model in Sec.3.4. We repeat this process for all the trajectories, and we get a dataset with sufficient knowledge for the offline RL to discover the optimal setting under each situation. Afterward, we compute the Reward-to-go for each trace at the end of the trace generation.

RobustMPC. Besides the Random demonstration trajectories, to accelerate the convergence of the decision transformer pre-training, we provide a portion of expert demonstration. We implement a simple RobustMPC-based[70] ABR controller and collect the running data, and we choose it because of its simplicity and reasonable performance.

Run-time inference After offline pre-training, the decision transformer model is ready to be used as the controller in a volumetric video streaming player. We set the observation scope $W = 10$, selected through an empirical study. We first run the bitrate adaptation controller with a RobustMPC-based online reinforcement learning agent, then after W time slot, we start predicting the a_{W+1} with decision transformer, and we move the history window forward by one-time slot. And continue the control in this fashion.

Algorithm 1: Decision Transformer for Volumetric ABR

Input: Task Set $\mathcal{T}$, random offline dataset $\mathcal{D}$, the expert demonstration $\mathcal{P} = \{\mathcal{P}\}$

for $n = 1, 2, 3, \ldots, N$ **do**
  for $T_i \in \mathcal{T}$ **do**
    for $m = 1, 2, 3, \ldots, M$ **do**
      Sample K-trajectory $\epsilon_{i,m}$ from $\mathcal{D}$;
      Find a demonstration $\epsilon'_{i,m}$ from $\mathcal{P}^i$;
      Combine Sample and Demonstration $\epsilon^{in}_{i,m} = [\epsilon'_{i,m}, \epsilon_{i,m}]$;
    Batch Assemble $G^Z_i = \{\epsilon^{in}_{i,m}\}^M_{m=1}$;
  Prepare the Training batch $H = \{G^Z_i\}^{|\mathcal{T}|}_{i=1}$;
  $\vec{a} = \mathrm{GPT2}_\theta(H)$;
  $\mathcal{L} = MSE(\vec{a}, a^*)$;
  $\theta = \theta - \beta \nabla_\theta \mathcal{L}$;

3.3.4 Frame Rate Adaptation

As shown in the motivation, the chunk-level ABR cannot react to the network fluctuation within the timespan of 5s chunk length L. While such mis-prediction could result in a physical injury for a immersive VR viewer, and should be removed as much as possible, we thus introduce a frame dropping mechanism to guarantee the low stalling in this scenario. The intuition of frame selection at the server side is to gracefully reduce the frame rate of a chunk when Client-Side ABR wrongly estimates the Bitrate. This can greatly resolve the stalling (re-buffering) problem since it will drop the tail part of a chunk when the playout deadline is reached. However, unlike HEVC, MPEG V-PCC's binary stream includes three separate streams, i.e., the geometry, attribute, and occupancy. This greatly complicates the frame dependency and thus needs a new model to depict this character. A new frame-dropping algorithm is needed to optimize the frame-dropping strategy. We propose the graph model for this dependency. We use an MCOP algorithm to find the optimal sending order for each GoP profile. Our algorithm runs offline and provides a profile for each

GoP structure (for each *ctc.cfg* file).

Insights: The frame dependency of MPEG V-PCC formatted volumetric video is more complex than 2D video. Each MPEG-VPCC chunk includes several GoPs, which start with a key frame (I-frame) and are followed by n non-key frame (P-frames). Each P-frame depends on previous frame $m_i \longrightarrow m_{i-1}$. There are two synchronized streams in a MPEG V-PCC chunk, one storing the texture (attribute) image, called AVD, and another storing the geometry (depth) image, called GVD, which includes two layers, one for the far side, and one for near side projection. The frame i needs geometry and attribute image to decode correctly. And each AVD and OVD follows standard 2D temporal dependency. We have several key observations on the V-PCC frame dependency:

- The playback must start from an I-frame.
- The tail frames of a GOP could be dropped with the neglectable cost of PSNR.
- The geometry far layer could be dropped.
- The attribute layer of a frame may be dropped with a small cost on PSNR.

We note that adding support to the IBBP GoP structure is possible by constructing a DAG graph on the streams. It is essentially similar to the GoP DAG of the IPP structure, the only difference is that the B frame could have both backward dependency to the previous P frame and a forward dependency on the next P frame, such forward dependency could slightly limit the possible frame dropping granularity. In this case, we usually need to drop the frames by entire P frame group (the two consecutive P frames and the B frames between them, the typical group includes four frames). Since IPP GoP is the primary structure for the low-latency streaming, so we focus on IPP structure here.

Graph Model Formulation of Frame Dependancy We encode these observations

into a directed acyclic graph $G = (V, E)$, as shown in Fig. 3.11. It is constructed as below:

Vertices. Each frame m_i includes three components, the near layer of a geometry map image $m_i^{g,n}$, the far layer of geometry map image $m_i^{g,f}$, and an attribute map m_i^a. Each component is represented by a vertex in Fig. 3.11, and the three components of all frames form the vertex set of G. We organize these vertices into three layers. The upper layer is the geometry near layer. The second layer is the geometry far layer. The third layer is the attribute layer. They are denoted as $\mathbb{M}_{g,n}$, $\mathbb{M}_{g,f}$, and $\mathbb{M}_a$. Now we add an auxiliary source vertex s and an auxiliary sink vertex t. Together they form the vertex set $V = \{s, t\} \cup \mathbb{M}_{g,n} \cup \mathbb{M}_{g,f} \cup \mathbb{M}_a$. Each vertex has two properties, value $p_q(\cdot)$, and cost $p_c(\cdot)$. Where $p_q(\cdot)$ must satisfy $p_q(\text{I-frame}) > p_q(\text{P-frame})$, and $p_q(\text{geometryNear}) > p_q(\text{attribute}) > p_q(\text{geometryFar})$. $p_c(\cdot)$ is directly available as the frame size in Bytes. Any assignment satisfying the above inequalities is valid.

Edges. Since our solution algorithm requires the values and cost on edges. We define the $p_q(\cdot)$ and $p_c(\cdot)$ of a directed edge $e = (v_a, v_b)$ to be $p_q(e) = p_q(v_b)$ and $p_c(e) = p_c(v_b)$. We now add edges. First, we add an edge from auxiliary source vertex s to each I-frame's geometry near layer, denoted as $\mathbb{M}_{g,n}^I$, that is:

$$E_s = \{(s, m_i^{g,n}) | m_i^{g,n} \in \mathbb{M}_{g,n}^I\}$$

Then we add an edge from each vertex in $\mathbb{M}_a$ to the virtual sink t, with value $p_q(\cdot) = 0$ and cost $p_c(\cdot) = 0$, so that algorithm can always drop all the consequent frames, denoted as E_t.

$$E_t = \{(m_i^a, t) | m_i^a \in \mathbb{M}_a\}$$

Now, we add two edges to link the three frame components of the same frame. $(m_i^{g,n}, m_i^{g,f})$ and $(m_i^{g,f}, m_i^a)$. To show the dependency that the geometry far layer of a frame depends on its near layer, and the attribute layer depends on geometry

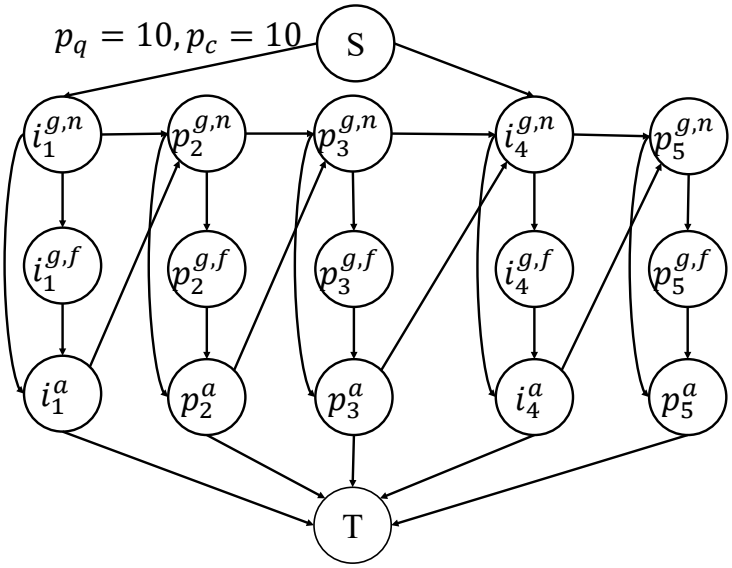


Figure 3.11: The frame unit graph.

layers. These edges are denoted as E_h. Finally, we link the attribute vertex m_i^a to the (i+1)th frames geometry near layer $m_{i+1}^{g,n}$.

Skip links We now model the insight that the far layer of a geometry map may be dropped. We simply add an edge $e = (m_i^{g,n}, m_i^a)$ so that there is a detour path from the geometry map's near layer to the attribute layer, skipping the geometry far layer. Similarly, we allow both geometry far layer and attribute layer to be dropped by adding an edge $e = (m_i^{g,n}, m_{i+1}^{g,n})$. We finish the graph construction here. This graph will differ if the GOP structure change. Thus, the chunk GOP structure (determined by the MPEG V-PCC profile) impacts its loss tolerance.

The Volumetric Frame Sorting Algorithm We can see that a cost-constrained optimal path on G encodes frame sequences that achieve the maximum PSNR while satisfying the bandwidth limits. Given the estimated bandwidth $\hat{b}$ and deadline t_d, the available network resources r is just the product of them, $r = \hat{b} \times t_d$.

$$r = \hat{b} \times t_d \tag{3.6}$$

Korkmaz et.al. [24] proposed the multi-constrained optimal path (MCOP) algorithm

Algorithm 2: The Volumetric Frame Sorting Algorithm.

Input: The profile set, B. The bitrate conditions, $\hat{B}$. The chunk length, t_d.

Output: Optimal frame order under each bitrate condition and profile, $\mathbb{S}^*$

```
Set 𝕊* ← [];
for t = 1, 2, 3, ..., T do
    Set 𝕊*_t ← [];
    for b̂ in B̂ do
        Compute r, given b̂ and t_d, using (3.6);
        Construct frame unit graph G for profile B_t;
        Compute the value of the p_q(·) for each edge of G;
        Run MCOP algorithm to solve G given r, S;
        𝕊*_t ← 𝕊*_t ∪ S;
    𝕊* ← 𝕊* ∪ 𝕊*_t;
```

Output: $\mathbb{S}^*$

to solve the optimal path problem on the acyclic graph. We use this sub-routine as a major building block in our Volumetric Frame Sorting Algorithm, which iterates through all the possible GoP structures listed in the available profiles and computes the optimal path under different bandwidth limits. This procedure is computed offline. While, in run-time, the server simply sorts the frames according to the corresponding frame order. The Volumetric Frame Sorting Algorithm is shown in Alg. 2.

The algorithm takes the profile set B, the ABR bandwidth steps $\hat{B}$, and the time slot deadline t_d as the input. The output is a lookup table containing the optimal frame order given a specific chunk encoding profile B_t and ABR bandwidth step $\hat{b}$. The algorithm iterates through T profiles, and for each profile, it iterates each bandwidth step $\hat{b}$.

Given the profile and bandwidth step, the algorithm computes the resources con-

Table 3.1: The Model Fitting Accuracy

Dataset	***Volumetric Sequence***	a_1^a	b_1^a	c_1^a	d_1^a	e_1^a	f_1^a	g_1^a	h_1^a	RMSE
8iVFBv2 [25]	**longdress**	30.42	-6.36	32.20	414.10	1.05	1645.88	756.59	1.49	0.0355
	loot	106.49	-78.55	106.99	-539.5	-3.78	-417.33	-1096.22	1.44	0.0466
	redblack	6.23	3.65	15.38	-565.3	382.62	-204.34	-28.34	0.34	0.0210
	soldier	1.62	8.95	11.15	-962.86	679.78	-383.76	-51.02	1.28	0.31
Owlii [69]	**basketball**	32.00	-5.70	14.54	141.46	0.01	80.79	106.60	1.15	0.1195
	dancer	32.00	-5.70	14.54	190.59	1.04	123.53	39.59	26.49	0.1067
	model	32.00	-5.70	14.54	229.56	0.01	108.39	79.33	1.17	0.0324
	exercise	32.00	-5.70	14.54	707.25	0.00	70.35	48.60	0.77	0.0569
MVUB [4]	**andrew10**	64.74	-39.53	43.26	40.13	0.00	121.25	-1.97	-0.23	0.0328
	david10	1.89	28.22	4.69	29.14	11.72	53.78	27.97	14.76	0.0181
	phil10	74.69	35.19	91.88	53.05	0.02	60.99	46.66	4.78	0.0357
	sarah10	32.00	-5.70	14.54	55.20	49.92	96.66	188.69	0.29	0.0282
	ricardo10	32.00	-5.70	14.54	123.04	15.55	186.21	-107.12	1.02	0.0495

straint r with the (3.6). Then we construct a frame graph, assign the edge weight, and run the MCOP [24] algorithm to compute optimal frame order as a path on G, denoted as S. The time complexity is $O(m + m\log(n))$ for each setting. In run-time, the server will choose the frame order from $\mathbb{S}^*$ given chunk profile t and the bitrate of the next chunk $\hat{b}$.

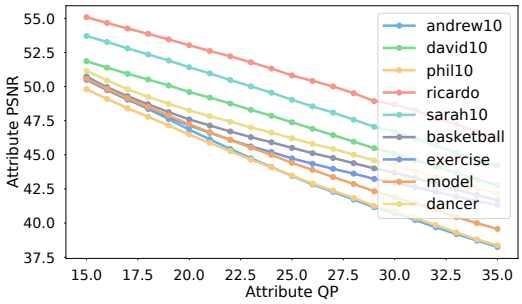


Figure 3.12: Attribute QP vs. PSNR-A

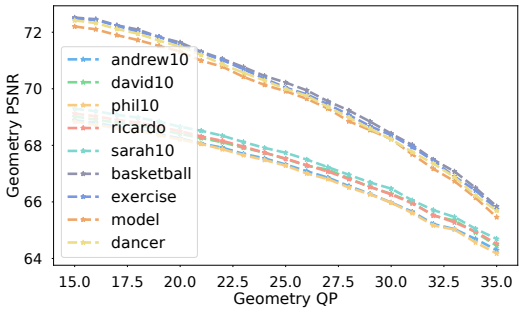


Figure 3.13: Geometry QP vs. PSNR-G

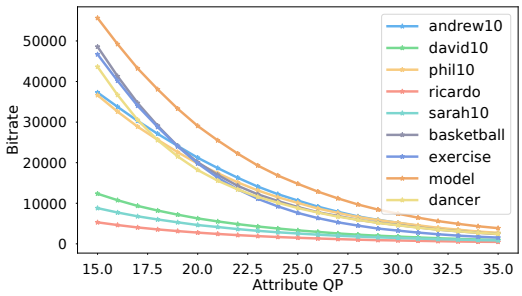


Figure 3.14: Attribute QP vs. BR-A

3.4 A New Rate-Distortion Model for Point Cloud Compression

Trade-off between quality and resource: In the adaptive volumetric video streaming problem, there is a tradeoff between the quality and the resource consumption, especially the network bandwidth. For example, tuning the quantization parameters to give a better quality will inevitably increase the bitrate of the resulting bitstream. If this rate is above the capacity of the network bandwidth, the playback of the volumetric video streaming would be interrupted, causing a freezing picture. To address this problem, we need to find a balance between the quality and the bandwidth resource consumption by tuning the quantization parameters. This problem is formally defined as the rate-distortion optimization problem.

As shown in Design overview, A MPD representation, see Fig. 3.4, will need a Rate-Distortion model to figure out the Bitrate and PSNR given the quantization parameter(QP) and downsampling rate (DR). However there is no such R-D model in the existing work, we propose the first in its kind model here. Due the to the similar codecs implementation between MPEG V-PCC and the H.264/AVC, we build our model based on Singhadia et.al.[55]'s work about Quantization Parameters and Meynets et al.'s work [36] on point cloud feature extraction. Singhadia et.al.[55]'s work established the relationship between the QP and the PSNR in a 2D scenario. While in the 3D scenario, simply considering the QP is not enough. Because the 2D image is dense and identically distributed. Contrarily, the 3D point cloud is sparse and unequally distributed over the coordinate system. Although it is reasonable to assume QP has an equal impact on each pixel for a 2D image, it is insufficient for point clouds. We found that the point density and curvature significantly impact the compression ratio (Bit per point). Therefore, we holistically consider point cloud density, point curvature, and Quantization Parameters in our new 3D rate-distortion model.

We first extract the point cloud-related intrinsic features of the input. The first one is point density, denoted as ϕ_p. This is the ratio between the number of points in a space and space volume. $\phi_p = \frac{|P|}{V}$. While point curvature is more complex. We borrow the definition from Meynets et al.'s work [36].

The curvature is defined on a neighborhood of a sampled point p. To compute it, one needs to first fit the surface around the neighborhood with radius h, denoted as $N(p, \frac{h}{2})$ using a quadratic fitting. Then using the coefficient obtained in the fitting, we can compute the local curvature centered in p.

$$\rho = \frac{(1+d^2)a + (1+e^2)b - 4abc}{(1+e^2+d^2)^{\frac{2}{3}}} \tag{3.7}$$

We define κ as the mean curvature over the entire point cloud. In practice, it is sufficient to use a sampled subset of points to estimate this value.

$$\kappa = \frac{1}{|P|}\sum_{p \in P} \rho_p \tag{3.8}$$

In contrast to ϕ_p, κ, which are intrinsic characteristics of the input point cloud, D_r is a tunable knob in most 3D libraries. Without losing generality, we consider uniform down-sampling here. In this setting, D_r is a linear knob to tune the effective point density ϕ_p^t, with relation $\phi_p^t = D_r\phi_p$. ϕ_p^t is an intermittent variable here, we replace it by $D_r\phi_p$ in our following discussion. After transforming effective point density ϕ_p^t to log scale, it has a sigmoid style jumping impact on the final PSNR and Bitrate. That is why we use a sigmoid function to depict this pattern. See (3.9).

$$\frac{g_1^a}{1 + e^{\lg(D_r\phi_p) - h_1^a}} \tag{3.9}$$

The G-PSNR function. The rationale behind this sigmoid logit relationship is rooted in the definition of point2point-PSNR(G-PSNR) [47]. For each decoded point f_i', the PSNR tool will find its matching point in the original point cloud f_i by searching the k-nearest neighbor in 3D space. So, as long as the point density is high

(in terms of a threshold, $\phi_p^t \geq h_1^{1/2}$), down-sampling will have a minimal impact on this searching process. But when the effective point density drops below a threshold $\phi_p^t < h_1^{1/2}$, it will massively interrupt this matching and cause a huge matching error, leading to a sudden PSNR drop. Together we conclude the form of the G-PSNR function as in (3.10):

$$\begin{aligned} f_{\text{PSNR}}^g(\text{QP}^g, D_r, \kappa, \phi_p) = a_1^g \times \exp\left\{-\left(\frac{\lg(\kappa)\text{QP}^g - b_1^g}{c_1^g}\right)^2\right\} \\ + d_1^g \times \exp\left\{-\left(\frac{\lg(\kappa)\text{QP}^g - e_1^g}{f_1^g}\right)^2\right\} \\ - \frac{g_1^g}{1 + e^{\lg(D_r\phi_p) - h_1^g}}. \end{aligned} \tag{3.10}$$

The Bitrate function. During the 3D-2D projection procedure, the 3D points are projected to several optimized patches. One can think of such a patch as an optimized projection plane that minimizes the magnitude of the projected RGB-D image pixel value. So, as long as the effective point density is above a threshold $\phi_p^t \geq h_2^{1/2}$, the mapping on the 2D patches will not change. So as long as $\phi_p^t \geq h_2^{1/2}$, changing ϕ_p^t has a limited impact on bitrate BR, so we also use the sigmoid function to model this pattern. On the other hand, point curvature κ also impacts the bitrate. As κ increases, the impact of QP will arise because a more complex geometry structure will need finer quantization on depth value. Since this impact changes only when κ greatly increase, we use a *log* scale function to depict this pattern, see $d_1^a \times \exp\left\{-\left(\frac{\lg(\kappa)\text{QP}^a - e_1^a}{f_1^a}\right)^2\right\}$.

$$\begin{aligned} f_{\text{BR}}^a(\text{QP}^a, D_r, \kappa, \phi_p) = a_2^a \times \exp\left\{-\left(\frac{\text{QP}^a - b_2^a}{c_2^a}\right)^2\right\} \\ - \frac{d_2^a}{1 + e^{\lg(D_r\phi_p) - e_2^a}}. \\ + f_2^a\kappa. \end{aligned} \tag{3.11}$$

$$f_{\text{BR}} = f_{\text{BR}}^a * (1 + m).$$

Our measurement found that the attribute(color) stream takes most of the encoding

bits. Hence, it is sufficient to assume the total bitrate is just $1+m$ times the attribute stream bitrate. The resulting **volumetric rate-distortion model** is then (3.12) and (3.11).

$$\begin{aligned} f^a_{\mathrm{PSNR}}(\mathrm{QP}^a, D_r, \kappa, \phi_p) = a^a_1 \times \exp\left\{-\left(\frac{\lg(\kappa)\mathrm{QP}^a - b^a_1}{c^a_1}\right)^2\right\} \\ + d^a_1 \times \exp\left\{-\left(\frac{\lg(\kappa)\mathrm{QP}^a - e^a_1}{f^a_1}\right)^2\right\} \\ - \frac{g^a_1}{1 + e^{\lg(D_r\phi_p) - h^a_1}}. \end{aligned} \tag{3.12}$$

Model Validation Dataset. We evaluate our model on a large corpus of volumetric video sequences, which includes three datasets, the MVUB [4], Owlii [69], and 8i [25], each including sequence with thousands of frames. They represent three different types of volumetric video contents. Microsoft Voxelized Upper Bodies(MVUB) is a dataset of human upper body, it is captured by four RGBD cameras, at 30fps, each lasts for 7-10 seconds. It is later processed to 1024x1024x1024 voxels. Owlii Dataset, on the other hand is full body human point cloud dataset, which includes four sequences basketball, dancer, exercises, and model, captured at 30fps, lasts for 20 seconds. Finally 8i, is a full body point cloud dataset that captured by 42 RGB cameras, including four sequences soldier, longdress, loot, and redblack, with 30fps frame rate and 10 seconds length. As these sequences has different content type, we think these sequences reflects generality of our proposed Rate-Distortion model.

Model Fitting. We use a standard curve fitting function in the scipy library to fit our model, with an random initial weight and a learning rate $\epsilon = 0.0001$. We set the maximum iteration to be 10k. The fitting parameters and the accuracy is listed below in Tab. 3.1.

Result. We can see in Table. 3.1 that the RMSE reached below 0.05 for all the sequences within the tested QP and Dr range. Therefore, our model is a practical building block for further rate-distortion optimization in the streaming framework. The fitting curve is demonstrated in the Fig. 3.12, Fig. 3.13, and Fig. 3.14. Where

the scatters are the true measurement values, and the solid curve is the fitted model predictions.

We also compared our 3D Rate-Distortion with three other approaches: 1. 2D-RD, that directly apply 2D rate-distortion to the three compressed V-PCC streams, (Attribute map, geometry map, and occupancy map); 2. MSE-RD, using a Mean-Square-Error(MSE)-based Rate-distortion model instead of PSNR-based Rate-distortion model; 3. E2E, that directly let the RL controller to determine the encoding parameters, without needing a Rate-Distortion model for Mapping between encoding parameter and bitrate.

In Fig. 3.15, we can see that VSAS-RD model has the best performance. Compared to the 2D-RD method, it considered the 3D related features, e.g., the point cloud density, which allows a more accurate R-D model on a wide range of different contents. Compared to MSE-RD approach, as MSE is not a good QoE indicator (as show MSE has a non-linear relationship to the user's subjective experience), therefore it has not achieved excellent QoE. Compared to the E2E-RD approach, As there are more unknown parameters for Decision Transformer to learn (it needs to learn Rate-Distortion model implicitly), this cause a higher learning complexity, and as a result leads to a slower convergence in pre-training, as well as slower adaptation to the new environment (it does not sure whether Rate-Distortion Relationship has changed for the new environment). This results in a worse performance and unstable controller behavior. The experiment confirms its inferior performance compared to the analytical 3D R-D model approach.

3.5 Evaluation

In this section, we perform experiments on various traces to evaluate the QoE performance of VSAS compared with several existing solutions. Our results on a wide

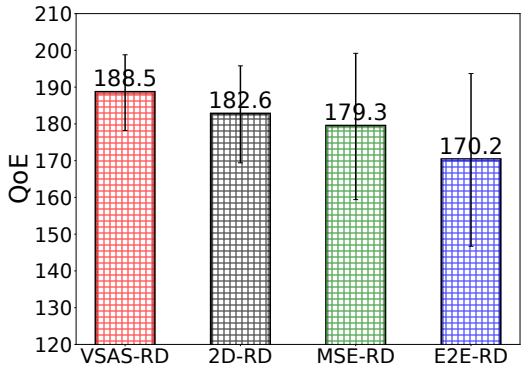


Figure 3.15: The performance comparison of different RD model on QoE

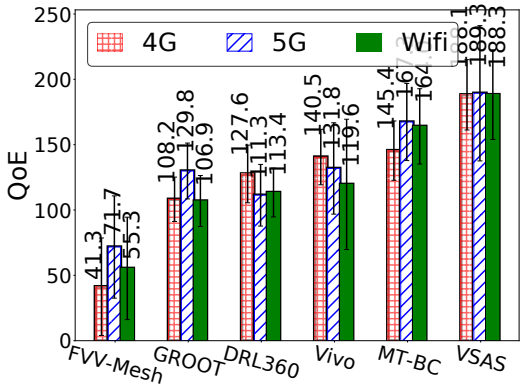


Figure 3.16: QoE Comparison

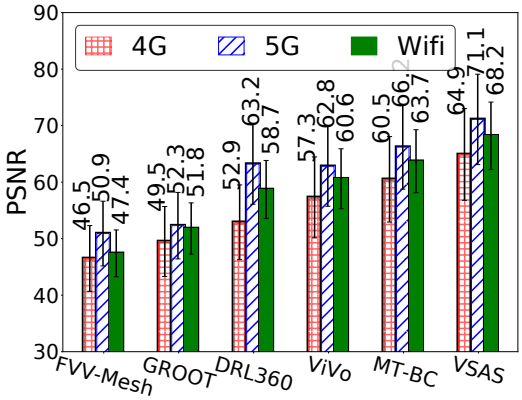


Figure 3.17: PSNR Comparison

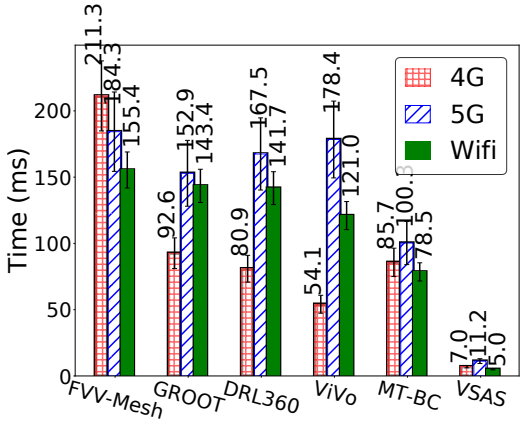


Figure 3.18: Average Stalling Per Chunk

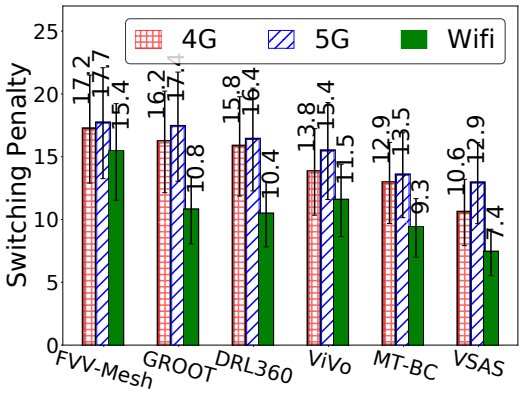


Figure 3.19: Quality Switching Penalty

range of open source volumetric video datasets show that the VSAS framework can avoid overshooting throughput and improve the QoE performance of VPCC under 4G, 5G, and Wi-Fi environments.

3.5.1 Experiment Setup

We cut the volumetric videos into the chunks of 2 seconds, and uses a bitrate sets of [1MB, 5MB, 7MB, 15MB, 25MB] according to the MPEG common test conditions for immersive video [40], the video encoder is the HEVC encoder used in the default encoder of the MEPG V-PCC's TMC3 [39], we use the *ctc-common.cfg* as the primary configure file, which is based on the IPPP GoP structure, and has a GoP size of 15. We slight modify this implementation with FFMPEG decoder, which has a much faster decoding speed than the default decoder used in the reference software. We use the GPT2-default (124M) as the foundation model of the decision transformer, and fine-tune it on our trajectory corpus with 35k sequences.

3.5.2 Methodology

Video traces. We built a large corpus of volumetric video sequences, which includes three datasets, the MVUB [4], Owlii [69], and 8i [25], each including sequence with thousands of frames. They represent three different types of volumetric video contents. Microsoft Voxelized Upper Bodies(MVUB) is a dataset of human upper body, it is captured by four RGBD cameras, at 30fps, each lasts for 7-10 seconds. It is later processed to 1024x1024x1024 voxels. Owlii Dataset, on the other hand is full body human point cloud dataset, which includes four sequences basketball, dancer, exercises, and model, captured at 30fps, lasts for 20 seconds. Finally 8i, is a full body point cloud dataset that captured by 42 RGB cameras, including four sequences soldier, longdress, loot, and redblack, with 30fps frame rate and 10 seconds length.

Network traces. We combined traces from three public datasets: Belgium 4G/LTE bandwidth dataset [61] for 4G/LTE, Raca dataset [48] for 5G/NR, and the Kaggle Internet Speed Dataset [21] for Wi-Fi. The network emulation is replayed by calling netem [31].

Baseline: We choose five baselines, FVV-Mesh [8], ViVo [12], GROOT [28], DRL360 [79], and MT-BC. They represent three state-of-art technologies for adaptive volumetric video streaming, see details as below:

- **FVV-Mesh[8].** We use MeshLab to compress original point cloud frames to mesh+texture atlas(FVV format). The Bitrate is determined by the number of sample points. We use the Ball Pivoting Surface Reconstruction method and choose 500, 1000, 2000, and 4000 sample points for four bitrate steps. And follow the system design in FVV-Mesh[8] to reproduce their results.

- **Vivo[12].** According to the setting in Vivo, we use Draco[9] to prepare the chunks. The Bitrate is determined by the leaf size of VoxelGrid. We set the leaf size of the VoxelGrid filter to 10, 7.5, 5, and 3.5 for four bitrate levels. The viewport-based pruning feature is implemented as mentioned in the paper.

- **GROOT [28].** GROOT is a G-PCC based volumetric video streaming system optimized for the mobile devices, e.g., smart phones. Their major contribution is to propose a parallel decodable tree data structure to compress the point cloud. And based on it, they achieved high-performance hardware acceleration on the Mobile GPU. They also consider the bitrate adaptation and viewport pruning problem. We use it as a baseline for mobile Volumetric video streaming system.

- **DRL360 [79].** DRL360 proposed a DRL framework to control the 360-degree video streaming. We compare it with our system by extending it to the volumetric environment.

- **MT-BC.** MT-BC is another offline reinforcement learning algorithm, which uses Multi-layer perception (MLP) instead of the GPT-2 as the sequence predictor. And it uses a behavior cloning as the imitation learning method, we replace Decision Transformer with Multi-task Behavior Cloning to see the impact of the decision transformer in our entire system.

- **VSAS.** Our newly proposed framework, with both chunk-level ABR and MCOP based frame dropping. We use 1k lines of golang codes to implement the server by modifying the HTTP server example of quic-go[7].

3.5.3 Overall performance.

Fig. 3.16 shows the QoE comparison among different baselines. We can see that VSAS outperforms the MT-BC, Vivo, DRL360, GROOT, and FVV-Mesh For 19.10%, 44.98%, 61.18%, 65.27% and 253.3%, respectively. It outperforms the MT-BC because MT-BC uses a Multi-layer Perception (MLP)-based backbone, which has a weaker ability than the GPT-2 used by the VSAS. VSAS outperforms the Vivo, as Vivo uses a traditional decoupled design that independently predicts the viewport and bandwidth, which has a smaller solution space, therefore limiting its potential. While the G-PCC compression format used by Vivo also has a lower compression efficiency. While GROOT is focused on speed instead of quality, it has a lower QoE due to its lower compression efficiency and empirical bitrate adaptation strategies. DRL360 uses a traditional online DRL ABR algorithm instead of the offline RL algorithm Decision Transformer we used. Such an algorithm has a slower convergence speed and sometimes have difficulties adapting to a new scenario, for example, a mixed video seqeuence from differnet dataset. Fig. 3.20 shows the QoE performance among different volumetric video sequences. We can see that VSAS outperforms the MT-BC, Vivo, DRL360, GROOT, and FVV-Mesh for 24.43%, 32.89%, 45.21%, 66.36%, and 213% percent on MVUB; which is similar to the result on 8i, which is 29.37%,

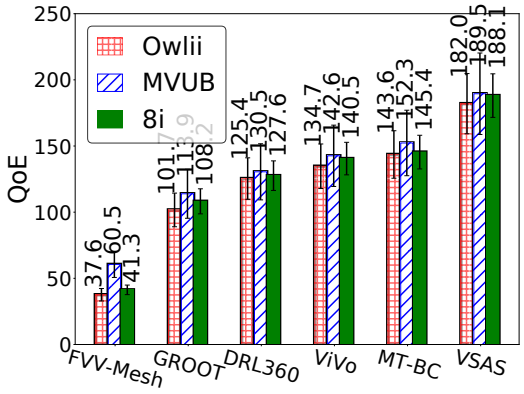


Figure 3.20: QoE performance on different volumetric video sequences

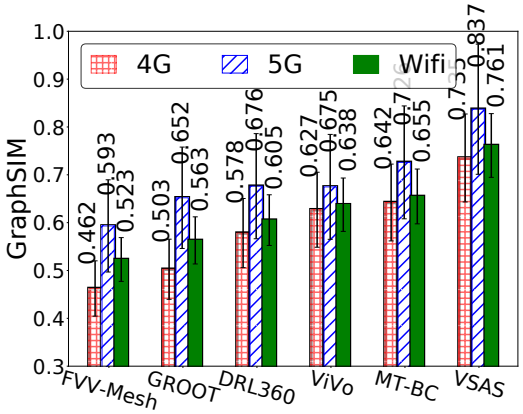


Figure 3.21: GraphSIM comparison on different networks

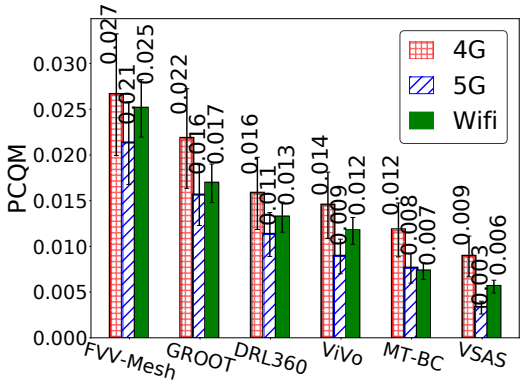


Figure 3.22: PCQM comparison on different networks

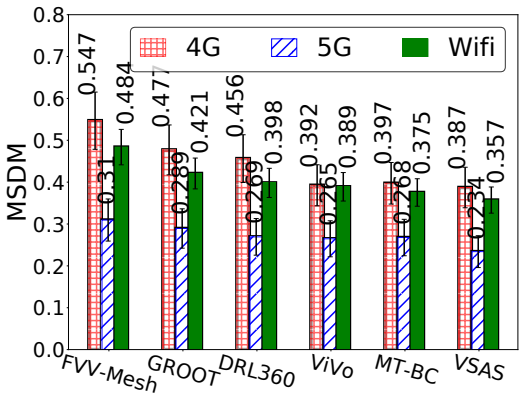


Figure 3.23: PC-MSDM comparison on different networks

33.88%, 47.41%, 73.84% and 355.4%, this is because MVUB has a more frequent content movement, giving a larger gain space for the frame dropping and Bitrate adaptation strategy.

3.5.4 QoE Breakdown.

Since different people could have different preferences over the QoE function, to reveal where the advantage comes from, we draw the breakdown figure for the three components of the QoE: the PSNR (quality), the stalling time (delay), and the quality

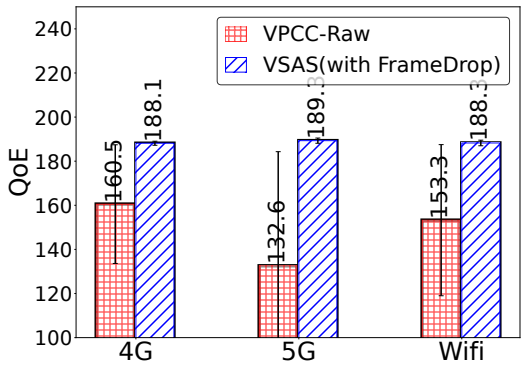


Figure 3.24: Impact of the Frame Dropping Mechanism

switch penalty(quality variation).

Quality. See Fig. 3.17, the PSNR of baselines. We can see that VSAS outperforms MT-BC, Vivo, DRL360, GROOT, and FVV-Mesh for 7.25%, 13.07%, 17.12%, 32.91% and 41.05%, respectively. The gain mainly comes from the higher compression rate of MPEG V-PCC, which achieves a higher quality under the same bitrate. Decision Transformer also gives a more accurate bitrate selection that maximize the utility of the avialable bandwidth, which translates to a higher quality. Note that in Fig. 3.21, Fig. 3.22, and Fig. 3.23, these three different video quality metrics demonstrate a similar result as of PSNR.

Stalling Time. While Fig. 3.18 shows the stalling time comparison, we can see that VSAS has significantly lower latency than all the baselines. This is mainly because of VSAS's frame-dropping mechanism that automatically drops the tail frames, which allows it to guarantee a very low stalling.

Quality Switching Penalty. Finally, regarding the quality switching penalty, see Fig. 3.19, where VSAS has a relatively lower quality switching penalty. This is mainly due to its higher bandwidth prediction accuracy and a more effective bitrate adaptation logic coming from the decision transformer, which has a much better control

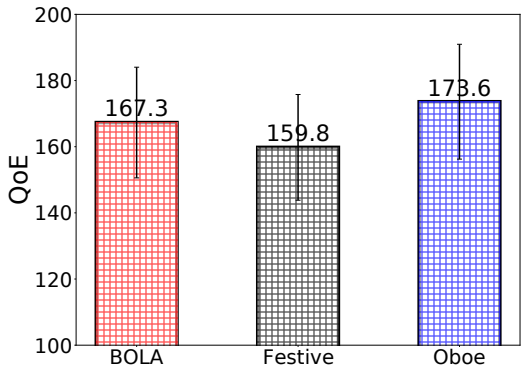


Figure 3.25: Impact of the Different Pre-fetching strategy

performance compared to the existing online reinforcement learning algorithms.

3.5.5 Ablation study.

We now explore the contribution of each module in VSAS. Fig. 3.24 illustrate the difference between VSAS with Frame Dropping on (VSAS-FrameDrop) and off (VPCC-Raw). We can see that VSAS-FrameDrop outperforms VPCC-Raw for 17.19%, 42.81%, and 22.84% improvement, respectively, under 4G, 5G, and Wi-Fi networks. This demonstrates the positive impact of adding the passive frame-dropping mechanism (step 2) to the traditional client-driven ABR.

Real-time behavior of the VSAS. To see the framd dropping in action, we draw the real-time bitrate plots in Fig. 3.28. In this trace, the ABR mechanism often leads to an exceedingly high bitrate or a sending rate far under the true bandwidth. This is because the decision granularity of Client-driven ABR is in the levels of 5 seconds. The prediction at the start of the time slot could be wrong within the 5s time interval. VPCC-Raw cannot adapt to this in-chunk network change. While seen in the 20-60s in the trace, the VSAS can dynamically decrease the Bitrate through frame dropping in a finer granularity, thus eliminating most of the stalling. (which happens when chosen Bitrate exceeds the true bandwidth). While VSAS is not perfect, when

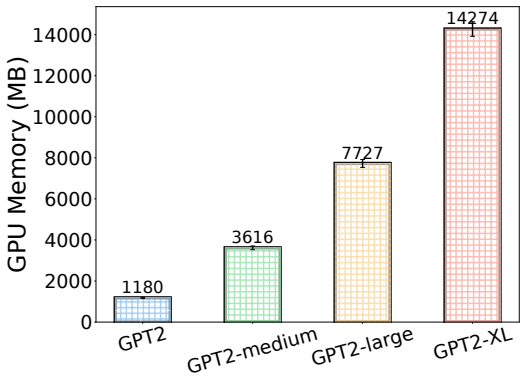


Figure 3.26: GPU Memory

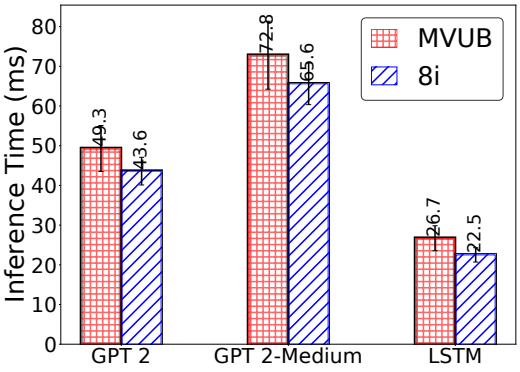


Figure 3.27: Inference Time

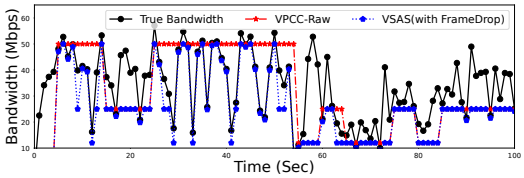


Figure 3.28: Real-time Bandwidth of VPCC-Raw and VSAS

true bandwidth exceeds the ABR chosen Bitrate, like between 60-80s, VSAS cannot increase the Bitrate above the VPCC-Raw chunk bitrate. Since frame dropping can only reduce the Bitrate of ABR given chunk, not increase, we leave this disadvantage to future work.

Impact of different Pre-fetching Methods. Fig. 3.25 explore the impact of different pre-fetching strategy on the VSAS, we replace the pre-fetching module of the Decision Transformer with BOLA, Festive, and Oboe, respectively. We can see that all three pre-fetching method achieve a similar QoE, at 167.3, 159.8, and 173.6, respectively. We conclude that VSAS can works with different pre-fetching algorithm without losing much of its performance.

The computation overhead. Our algorithm uses GPT-2 as the foundation of the sequence predictor, however, GPT-2 is a large model, we measure the com-

putation overhead of the GPT-2. Since the Chunk size is 1-5 seconds for most applications(Video-on-demand, Live-streaming), so as long as the inference time is lower than the 1000ms, than the system could achieve a real-time control. See Fig. 3.27, the inference time of the different versions of the Decision Transformer running on different versions of GPT-2 model(default, Medium), we can see that GPT-2 default version costs 49.3ms on MVUB video sequences, and 43.6ms on 8i sequences, this difference comes from different input token length. while GPT 2-Medium takes 72.8ms and 65.6ms respectively, which is longer than the GPT 2 default as it includes more parameters. While, we also involve a LSTM-based version for comparison, which is also called Decision-LSTM, it has a inference time of 26.7ms and 22.5ms. However, note that all these inference time is far smaller than the real-time threshold(1000ms), and therefore, we prefer the large model GPT2 that fits into our system for better performance. The memory costs is show in Fig 3.26. This size is able to fit into the GPU memory of most of the commodity machines, so we consider this system practical for deployment.

In conclusion, the VSAS outperforms state-of-art systems by a large margin. The improvement mainly comes from better client-side ABR algorithms and the ability to optimally drop the frames when the network experience a sudden fluctuation.

3.6 Discussion and Conclusion

This paper proposed a new DASH-based framework for volumetric video streaming, supporting MPEG V-PCC codecs. This framework features a higher temporal compression ratio and a better 3D QoE. We observe stalling and smoothness as the major difference between existing 2D system and 3D systems. To tackle this challenge, we proposed a new rate-distortion model for 3D QoE, and a new frame dropping mechanism tailored for MPEG V-PCC, we control the bitrate adaptation through a new offline learning paradigm, and develop a decision transformer-based ABR controller.

We implemented our framework, and evaluated it on a large dataset, the results shows a 1.67x improvement over three SOTA systems. Although our work outperforms the existing baselines, it still has potential room for further improvement, in terms of handling the scenario change during the playing, and faster server-side encoding to make this framework suitable for live volumetric video streaming.

Chapter 4

Few-shot Adaptive Bitrate Volumetric Video Streaming with Prompted Online Adaptation

4.1 Introduction

In the era of virtual reality, a new type of application volumetric video streaming demonstrate a good potential. It provides a free-viewport experience, which allows not just the rotation of the viewing point but also the changing of viewing position. It is now feasible to capture the volumetric video in real-time with off-the-shelf devices. For example, three Kinect cameras can support full-angle volumetric video, while one is sufficient for one-angle video streaming. At the same time, there are more and more VR/XR viewing devices at consumer exposure. For example, Apple Vision Pro has been widely considered the most innovative product in recent years, along with a wide range of XR glasses from other vendors [60, 13].

There have been existing works in supporting tile-based volumetric video streaming. These works mainly focus on optimizing the volumetric video streaming by several

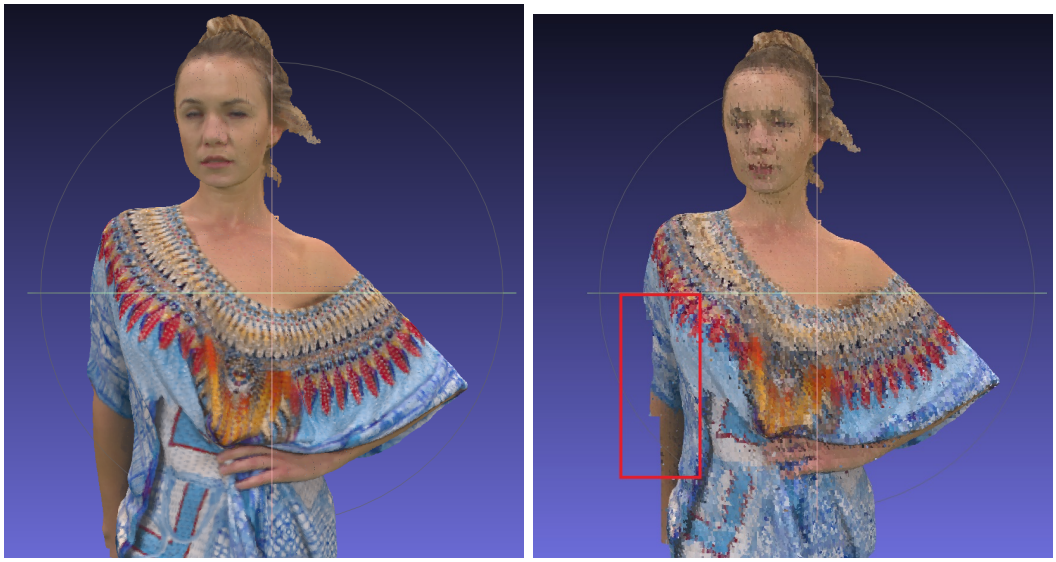
viewport-based heuristics using a deep reinforcement learning framework [29]. Where they train a policy to optimize the system performance under a given distribution. However, in practice, these RL-based systems face two challenges: first, the poor generalization ability to the unseen environment, that is, when the system encounters a strange environment, its performance will degrade greatly; second, the catastrophic forgetting, that is when an RL policy is trained on a mixed diversified dataset, it could forget the knowledge, and perform badly on a single dataset.

Recently the Large Language Models have led to a new trend of the AGI (Artificial General Intelligence). Researchers found that as the size of the language model increases, its ability to few-shot learning shows a steep improvement. ChatGPT [3] can learn a new task without retraining the core transformer weights. This provides us with a new approach to solving the above-mentioned out-of-distribution adaptation problem by reformulating the volumetric video streaming control into a sequence prediction problem. We can borrow LLM's ability to achieve a few-shot adaption for new datasets, thus improving the generalization of the tile-based volumetric video streaming systems.

Designing a few-shot offline reinforcement learning system for volumetric video streaming systems is non-trivial. The first challenge is a lack of a framework supporting edge-assisted few-shot reinforcement learning. The second challenge is modeling the volumetric video streaming system into a multi-variate sequential prediction problem. The third challenge is training such a model and balancing the online adaptation ability with the inference cost. To tackle them, we introduce a systematical design of a novel GPT-based model demonstrating a high few-shot learning ability in our prototype implementation. We evaluate it on a large dataset corpus, and it shows a fast run-time adaptation and good out-of-distribution generalization.

We summarize our contributions as below:

- For the first time, we study the generalization problem of the volumetric video



(a) Accurate Prediction (b) Rotation Error

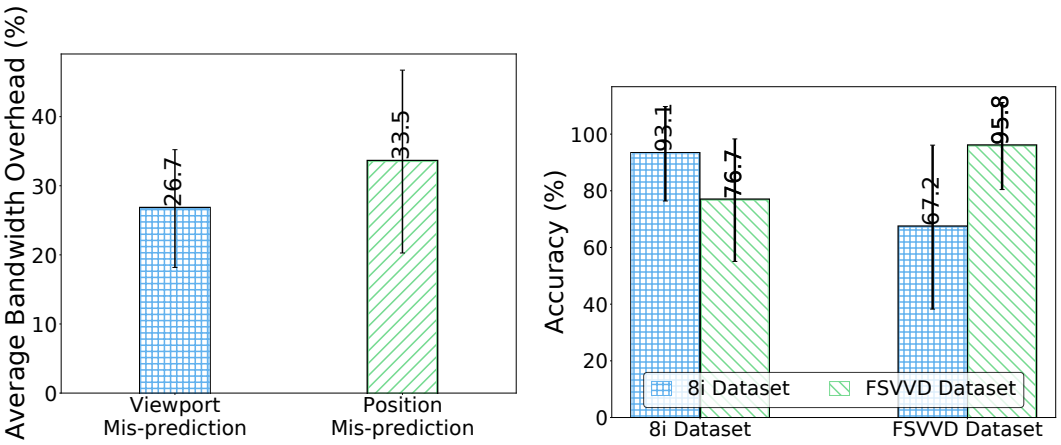
Figure 4.1: Visual Effect of Viewport Prediction Error.

streaming system through a measurement study.

- Based on these observations, we design the first generalized few-shot volumetric video streaming (FewVV) framework, and reformulated the volumetric videos streaming control as a multi-variate sequence prediction problem.

- We solve this problem using a GPT-based transformer Volumetric Causal Transformer.

- We evaluated FewVV on various networks, content, and viewing environments. We show that FewVV outperforms the existing system on overall QoE and can generalize to the out-of-distribution video sequences effectively.



(a) bandwidth overhead

(b) Rotation Error

Figure 4.2: Viewport Prediction Generalization Issue

4.2 Related Works

4.2.1 Volumetric Video Streaming Systems

Volumetric Video Streaming Systems are systems that support the immersive 6 Degree-of-Freedom Video Streaming. Different from 360-degree videos, volumetric video streaming allows not only the rotation of the viewport but also the movement of the viewing position. Such a video format has been widely used in movie production [52], virtual reality entertainment [71], and industry's remote technical support [73]. These systems can be classified into three categories according to their storage format, the point cloud-based [12], the mesh-based [8], and video-based [39]. More recent development has been presented in Neural Radiance Rendering(NeRF) [37] and Neural Network-based methods [45]. Vivo [12] is one of the earliest research on the tile-based point cloud format video streaming. They first demonstrate the components of the volumetric video streaming systems, then propose a *visibility-aware* streaming system. Vivo [12] are different from our work, as we study and explore the need for generalization, instead of the basic viewport heuristics. GROOT [28],

on the other hand, is the first work that proposes a new tree data structure that is effectively parallel, which allows GPU-assisted decoding, which hugely accelerates the frame rate of the decoding process, their implementation achieves a piratical system working on a wide range of mobile devices (smartphones). Our work does not focus on the codecs of the dynamic point cloud and, therefore is agnostic to GROOT [28]. Zhang Ding, et al. [75] considered the case of multi-user streaming with Wi-Fi 6's beamforming and multi-casting technique to improve the wireless speed. They consider a cross-layer design to take the viewer's movement and relative positions caused occlusion into bandwidth scheduling. Our work differs from it as we consider movement for scheduling the content in the application layer, while their work uses the user's moving trajectory to improve link layer efficiency. The focus of our work is on the generalization of the volumetric video streaming systems, the foundational architecture belongs to the research stream of Vivo, the tile-based volumetric video streaming with point cloud format. Now we analyze the generalization problem in existing systems and inspire our novel design.

4.2.2 Visibility-aware adaptive volumetric video streaming

Visibility-aware adaptive volumetric video streaming is a approach that first cut the point cloud into a number of tiles, each tile is compressed and transmitted independently with its own compression parameters. Because the visibility of each tile are different from others, they have different visual importance, for example, if some tiles are far from the viewer, they are less important, while if a tile is occluded by other tiles, it could be excluded from transmitting. The visibility-aware adaptive volumetric video streaming use the visibility to compute the relative importance of the tiles and choose different transmission parameters accordingly.

4.2.3 The Generalization Problem in Volumetric Streaming Control

The impact of viewport and position (FoV) prediction error The existing volumetric video streaming systems depends heavily on the viewport prediction, however, it could be difficult to achieve high accuracy on the practical environment [34, 74]. Which has largely undermined the effectiveness of the viewport pruning-based tiled volumetric streaming [12]. In Fig.4.1, we depict the visual impact of the viewport prediction error. Where Fig.4.1(a) is the accurate volumetric video of the dancer, the viewport predictor in Fig.4.1(b) failed to predict the rotation angle accurately, thus it decide to prune the tile including the dancer's lower limb, causing a fracture of picture. To quantify this loss, in Fig.4.2(a), we show the bandwidth cost of rotation (type 1: yaw, pitch, raw error) and position (type 2, x, y, z coordinates error) error for the volumetric video streaming system.

The FoV prediction generalization The Field-of-View (viewport) prediction of existing works takes a multi-layer-perception (MLP) approach, which is a traditional neural network-based method, although this approach has strong performance on the training dataset, it has difficulties to generalize to the out-of-distribution (OOD) environment. We demonstrate this effect in Fig. 4.2(b), when MLP is trained on 8i dataset, and used on FSVVD [14] dataset, its accuracy drops from 93.1% to 67.2%. Similarly, when MLP is trained on FSVVD dataset and used on 8i dataset, its accuracy drops from 95.8% to 76.7%. To resolve this problem we need a sequence model that can generalize to the new environment in a few-shot basis (tuned by 5-10 expert examples). Such capability has been noticed in the recent large language models. In this paper, we try to introduce this potential approach in this paper. We first design a new framework to facilitate few-shot adaptation of the volumetric video streaming system.

4.3 The GVVS framework

The system mainly includes three parts. The **Volumetric Sequence Server** (VoD) running on the Edge Server; the **Unity Player**, a player application on the XR headset, that receives the point cloud tiles and renders them to the environment; and the **Birate Selector**, whose job is to control the bitrate according to the feedback coming from the Unity Player. Now, we introduce them in detail and show their interaction workflow.

4.3.1 Volumetric Sequence Server

Volumetric Sequence Server has following modules:

Storage of the DPC The dynamic point cloud (DPC) is stored on the edge server using lossless octree-based compression. We use Draco Library [9] from Google as our encoder. This design allows a Video-on-Demand service, and can be easily extended to the Live-streaming scenario.

Tile cuts and compression The dynamic point cloud (DPC) is a series of frames. Each frame is a static point cloud, a set of unordered points in space. Volumetric Sequence Server first decompresses the lossless full DPC into the raw format, then sub-divides the space into N blocks, and each block is called a *Tile*. The tiles are processed independently using Draco [9], an octree-based encoder, which can compress the Tile into a target bitrate. It then transmit these compressed tiles in a data stream to the *Player*.

4.3.2 Unity Player

The Unity Player is an application running on XR devices (e.g., HoloLens 2 [60], Oculus Quest II [13]) written using Unity Game Engine [11]. It is the primary component on the receiver side and is directly responsible for rendering and showing the DPC to the viewers. Its function is in three folds: First, it collects the viewers' viewport information, history bandwidth, and other data; Second, it sends the information to the *Bitrate Selector* to ask for a proper bitrate selection vector a_{t+1} for next frame $t+1$, then the *Bitrate Selector* will send the vector a_{t+1} to the Volumetric Sequence Server (Edge Server); Finally, the Unity Player receives the tiles compressed by Draco [9] from the Server, and render it on the XR device.

4.3.3 Bitrate Selector

The Bitrate Selector is located on the Edge server, that receives the feedback signals (viewport, distribution shift, bandwidth) from the Unity Player, and runs the volumetric causal transformer algorithm to output a bitrate selection action. It reuses the history prompt when there is no distribution shift, while when there is a shift, it attaches the extra prompt from the *Unity Player* and adapts to the new scenario using a few-shot adaptation. We give the details of the Bitrate Selector Algorithms in Sec. 4.5.

4.4 Problem Formulation

QoE Model

Quality-of-Experience (QoE) is a metric that reflects the human's subjective perception score over multimedia content, there are two factors that impacts this perception: first, the quality of the frame q_j; second, the playback latency (delay) l_j. Since we

have tiled the frame into several tiles, the quality of these tiles can be independently adjusted. We denote the quality of the i-th tile in the j-th frame as q_j^i. This quality is proportional to the bitrate allocated to this tile $q_j^i \propto b_j^i$. We define the quality of the frame j to be the average of each tile. Let the number of chunks be $\mathbf{C}$. See Eq. 4.1.

$$q_j = \frac{\sum_{i=1}^{\mathbf{C}} q_j^i}{\mathbf{C}} \tag{4.1}$$

And QoE of the frame is the linear combination of the quality and stalling time, that is

$$QoE_j = a_1 l_j + a_2 q_j. \tag{4.2}$$

The instance value of the hyper-parameter a_1 and a_2 is different between different viewers, according to their preferences.

The GVVS Bitrate Selection (GVVS-BS) Problem

We now formulate the GVVS bitrate selection GVVS-BS problem. First, the objective of the system optimization is to maximize the accumulated QoE of all frames across all tiles. The action knobs are the bitrate b_j^i for each tile i in each frame j. The constraint is the dynamic bandwidth limits B_t, and the available bitrate ranges $\mathcal{R}$. The formulation is shown as follows:

$$\begin{aligned} &\arg\max_{b_j, I(c_j)} \sum_{j=1}^{J} QoE_j \\ &\text{s.t. } B_j \geq \sum_{i=1}^{M} I(c_j^i) b_j^i \\ &M \geq \sum_{i=1}^{M} I(c_j^i) \\ &b_j^i \in \mathcal{R} \end{aligned} \tag{4.3}$$

Where the algorithm should choose the bitrate b_j^i for each tile j in each frame j, and determine the value of indicator function $I(c_j^i)$ whether a tile i should be transmitted.

Clearly, since the system works under a VoD assumption (the video is pre-compressed into several bitrate levels). So the available bitrate b_j^i must belong to a discrete set of the bitrates $\mathcal{R}$. The sum of the transmitted bitrate for each tile $I(c_j^i)b_j^i$ must not exceed the available network bandwidth.

Problem Analysis. The GVVS-BS problem includes three sub-problems: first, the viewport prediction, which predicts the viewport of a user in a scene-given given content; second, the bandwidth prediction, which uses past bandwidth and buffer status to predict the future bandwidth; third, the bitrate allocation, where the available bitrate is allocated to the proper tiles that matter to the user. Some existing systems solve these three problems in an agnostic manner. However, these three problems are not independent. For example, the bitrate selection could change the user's viewport in the next time slot, which sometimes leads to a vicious circle of fluctuation. On the other hand, because the viewport could change greatly according to different users' behavior characteristics and their interest in different content, the out-of-distribution situation is common in real-world deployment. Such a generalization problem is a major obstacle to an ideal and practical visibility-aware streaming system.

Potential Approach. To address the first challenge, the error propagation due to the decoupled design of three modules, we propose an end-to-end sequence modeling approach to directly predict the action of the next time slot, given the history of previous states, actions, and rewards. In this way, the three subproblems are simplified into a sequence prediction problem. To further improve the generalization ability and address the second challenge of out-of-distribution adaptation, we design a few-shot learning approach on top of the sequence model, which uses trajectory prompts to hint at the sequence model during the run-time, allowing a fast online adaptation. In the next section, we give a detailed description of our design.

4.5 End-to-End Causal Transformer for GVVS

4.5.1 Preliminaries

Formally, reinforcement learning is a sequential decision-making process. The algorithm (agent) takes feedback from the environment and outputs the action according to a learned policy, hoping to optimize an objective. In traditional settings, reinforcement learning is operated online, where the agent explores the environment by trying some randomly sampled actions, the environment feedback to the agent with an instance of reward R and the state transition $\mathcal{F}$, and the RL agent slowly learns the state transition, while solving an optimal policy under this environment.

However, the RL problem can also be formulated to a offline scenario, where all the history is recorded as trajectories of actions, reward, and states, these trajectories are collected into a dataset $\mathcal{D}$. A K-trajectory is defined as a sequence of length K, $\{r_0, s_0, a_0, r_1, s_1, a_1, \ldots, r_{K-1}, s_{K-1}, a_{K-1}\}$, that is a sample of a teachers action sequence. The offline RL should learn the optimal policy solely depends on this dataset's trajectory.

States For the GVVS-BS problem, we first need the information to predict the bandwidth and the viewport, for bandwidth, we choose the bandwidth of past W time slots as the states $\mathcal{B} = \{B_{t-W}, \ldots, B_{t-1}, B_t\}$, we also define the N_q available bitrate levels for each tile to be $\mathcal{Q} = \{q_1, q_2, \ldots, q_{N_q}\}$. For viewport, each viewport point is defined by a 6-tuple $v = [x, y, z, yaw, pitch, row]$, and we use a W time slots viewport history $\mathcal{V} = \{v_{t-W}, \ldots, v_{t-1}, v_t\}$ as the viewport state. We also include the buffer occupancy $\mathcal{O}$ in our states. The state vector is then $\vec{S} = [\mathcal{B}, \mathcal{Q}, \mathcal{V}, \mathcal{O}]$.

Actions For each time slot, we determine the bitrate of each of M tiles, according to the states provided above. We also set an indicator action, that indicates whether

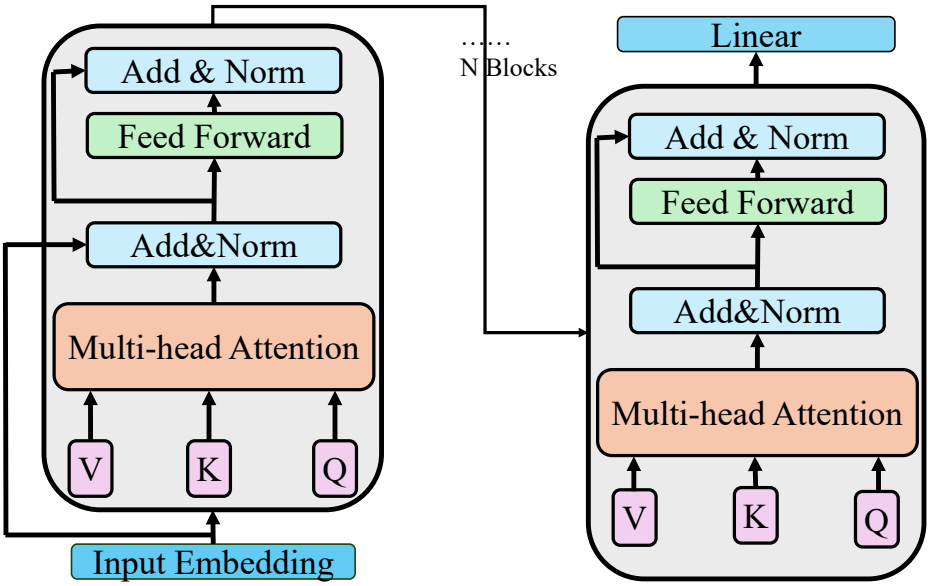


Figure 4.3: The architecture of volumetric causal transformer model.

a tile will be included in the transmission. That is bitrate selection vector $\vec{B} = [b_1, b_2, \ldots, b_M]$, and the tile occupancy vector $\vec{I} = [I_1, I_2, \ldots, I_M]$, where $I_i \in [0, 1]$. The action vector is the combination of these two actions $\vec{a} = [\vec{B}, \vec{I}]$.

Reward After each round, the feedback reward is computed as the instant QoE, which is the interpolation between the quality, and the latency shown in Eq.(4.2)

Joint Optimization of Higher-order data. As seen in the above formulation, the order of states and actions is higher than a traditional 2D ABR problem, where the action only has one dimension, and states come from 1-dimensional embeddings. Higher-order data challenges a traditional multi-layer-perception (MLP) neural network to capture their internal relationship. To address this problem, we instead use a stack of decoder-only transformer modules with many parameters to solve this complex joint optimization problem and train it on a large-scale trajectory dataset. See the details in the next section.

Algorithm 3: Volumetric Causal Transformer Pre-training

Input: Scene Task$\mathrm{T}_{\mathrm{scene}}$, Network Task$\mathrm{T}_{\mathrm{network}}$, the history training trajectory dataset $\mathcal{D}$, and the few-shot expert prompt dataset $\mathcal{P} = \{\mathcal{P}_{\mathrm{scene}}, \mathcal{P}_{\mathrm{network}}\}$

for $n = 1, 2, \ldots, E$ **do**
  Initialize the batch pool;
  for $T_i \in \mathrm{T}_{scene}$ **do**
    for $m \in \{1, 2, \ldots, Z\}$ **do**
      Randomly choose a trajectory $\eta_{i,m}$ with length K in $\mathcal{D}$;
      Generate the prompt $\eta'_{i,m}$ from $\mathcal{P}^i_{\mathrm{scene}}$;
      Form the input sequence $\eta^{in}_{i,m} = [\eta'_{i,m}, \eta_{i,m}]$;
    Summarize the scene task batch $S^Z_i = \{\eta^{in}_{i,m}\}^Z_{m=1}$;
  for $T_i \in \mathrm{T}_{network}$ **do**
    for $m \in \{1, 2, \ldots, Z\}$ **do**
      Randomly choose a trajectory $\eta_{i,m}$ with length K in $\mathcal{D}$;
      Generate the prompt $\eta'_{i,m}$ from $\mathcal{P}^i_{\mathrm{network}}$;
      Form the input sequence $\eta^{in}_{i,m} = [\eta'_{i,m}, \eta_{i,m}]$;
    Summarize the network task batch $L^Z_i = \{\eta^{in}_{i,m}\}^Z_{m=1}$;
  Integrate a training batch $A = \{L^Z_i, S^Z_i\}^{|\mathrm{T}_{\mathrm{scene}} \cup \mathrm{T}_{\mathrm{network}}|}_{i=1}$;
  $\vec{a} = VCT_\theta(A)$;
  $\mathcal{L} = MSE(\vec{a}, a^*)$;
  $\theta = \theta - \beta \nabla_\theta \mathcal{L}$;

4.5.2 Model Design

We design the volumetric causal transformer architecture in Fig.4.3, which is based on the decoder only transformer architecture of GPT [3], and we introduce its detail as below:

The decoder block: The overall architecture of the volumetric causal transformer

is a stack of basic transformer decoder block. The input is first mapping into an embedding, and then feed into the first layer of the causal decoder block. The block uses a multi-head attention as the core, followed by a feed forward layer.

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^T/\sqrt{d_k})V \tag{4.4}$$

We give the equation of the attention in Eq.(4.4). where d_k represents the scale factor. This stack of decoders include a large amount of the weights, and is able to learn a complex knowledge during the pretraining given a large dataset, which allows it to have a better generalization ability than the existing approaches.

4.5.3 Training Process

During the training, we choose two sets of datasets with several training tasks $\mathrm{T}_{\text{train}} = \mathrm{T}_{\text{scene}} \cup \mathrm{T}_{\text{network}}$, the test tasks T_{test} are mixed tasks with different combination of the different network and scene. For each task, we generate a prompt. Then, the generated prompt is attached to the head of the raw decision sequence. Let the prompt length be K^*, and the raw decision sequence of length K, then the attached input sequence for the training is of length $K + K^*$. These sequences are organized into batches before the training.

See Alg. 3, the Volumetric Causal Transformer pre-training. The training is done in batches. There are two levels of the batch: first, the outer batch, which groups E training instance, and second, the inner task batch, which chooses Z different examples from each task (network task, scene task). We first sample a trajectory $\eta_{i,m}$ from the random offline dataset $\mathcal{D}$, which includes a mixed large dataset generated with rule-based and random action trajectories. Then we generate the prompt $\eta'_{i,m}$ from the expert demonstration dataset $\mathcal{P}^i_{\text{network}}$ for network tasks, and $\mathcal{P}^i_{\text{scene}}$ for scene tasks. Then, the prefixed input sequence $\eta^{in}_{i,m}$ is formed by prefixing the prompt to the main input sequence. The sequence then has a length of $K + K^*$. Then, we

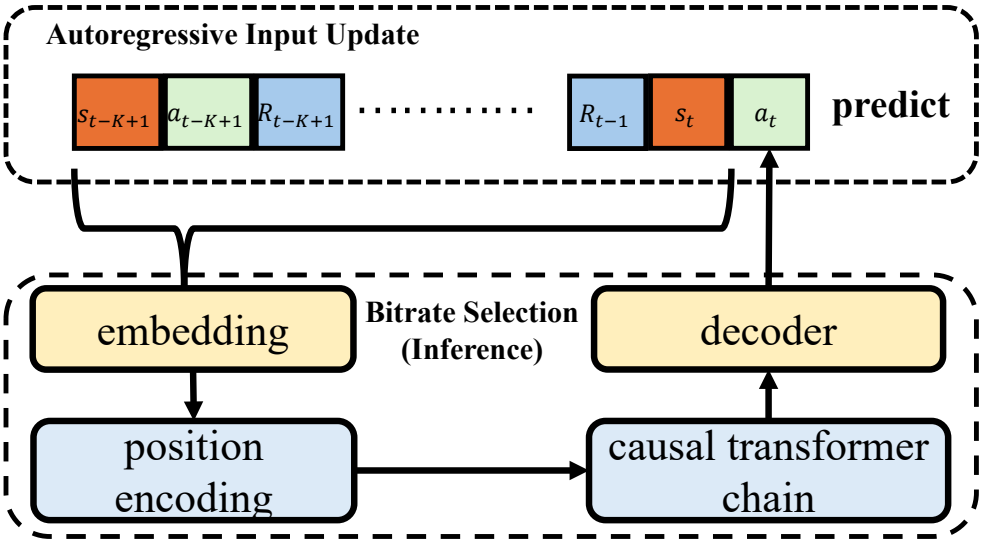


Figure 4.4: The inference pipeline of GVVS controller in run-time

group Z traces to form the scene task batch S_i^Z and network task batch L_i^Z for each instance of the tasks in the task categories.

The training is done by first attach both scene batch and network batch to a whole epoch batch $A = \{L_i^Z, S_i^Z\}_{i=1}^{|\mathrm{T_{scene}} \cup \mathrm{T_{network}}|}$, then the optimizer first do a forward step $\vec{a} = VCT_\theta(A)$, where VCT model outputs the next action vector $\vec{a}$. The MSE loss between $\vec{a}$ and a^* is computed $\mathcal{L}$, and the parameters of the causal transformer θ is updated through back-propagation with a learning rate β.

In this way, the Volumetric Causal Transformer is trained to recognize the distribution shift (task change) directly by recognizing the prompt. It can even adapt to the scenes and network from our training dataset, We confirm this ability and its limitations in our ablation evaluation Sec. 4.7.

4.5.4 Inference with Prompt for Runtime Adaptation

During the test time, as the Volumetric Causal Transformer is trained to take prompt and the input trace, we first initialize a zero input trace at the beginning of the test(real-world system operation). Then, as the data accumulate, we can get a full-length K-input sequence in the real world, which is maintained in an auto-regressive fashion. The receiver controller collects and generates the prompt, including an action sequence and high-accuracy rewards. The process is seen in Fig. 4.4. The system can then work by repeating the line. 16 in the Alg. 3 without computing the loss and the back-propagation process. Prompt-based tuning does not require the model parameter θ to be fine-tuned online, and thus has a faster adaptation speed and requires less computing power during the deployment.

4.6 Implementation

4.6.1 Unity Player Implementation

We implement GVVS *Unity Player* modules by extending Draco's example on Unity Engine [11] with a Microsoft Mixed Reality Toolkit [43]. The player is responsible for receiving and buffering the received point cloud frames into a playback buffer, then it decodes the point cloud frame and concatenates the point cloud tiles together after depression, then it is fed into the rendering queue. The player then renders the point cloud onto a stub object with color using the Unity Pcx-plugin [41]. The player runs a small module that detects the context change event and feeds the volumetric sequence server with high-quality feedback when there is a change in scene or people.

4.6.2 Volumetric Sequence Server Implementation

We use PyTorch to implement Volumetric Causal Transformer(FewVV). We train FewVV on our medium-quality dataset, we collect the expert data using the converged PPO agent, and we then mix in a large portion of random demonstrations to expand the size of the dataset. We implement the Storage Module using the quic-go example HTTP server [7], we implement the tile compression and decompression module using Draco [9], and implement the tile cut with Open3D Library [81].

4.7 Evaluation

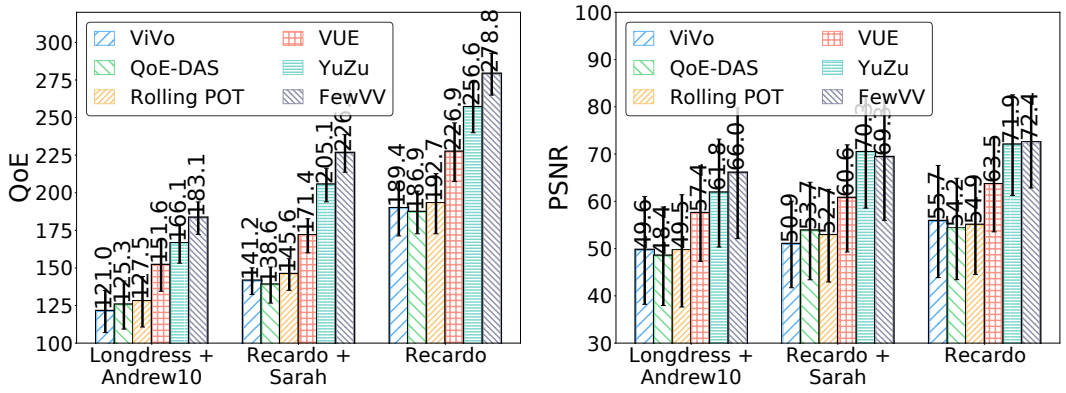
4.7.1 System Setup

Video Parameters: Our video sequences are chosen from the database of 8iVSLF [25], MVUB [4], and Owlii [69]. We cut the original frames into 12 tiles referencing the setting in Vivo [12], according to the 3D space boundary of each frame.

Network Trace: Our network traces are mainly from dataset Belgium 4G/LTE bandwidth logs (bonus) [61], Kaggle Internet Speed Dataset [21] and a 5G dataset [48], we divided these datasets into 4G/LTE, 5G, and WiFi for trace generation.

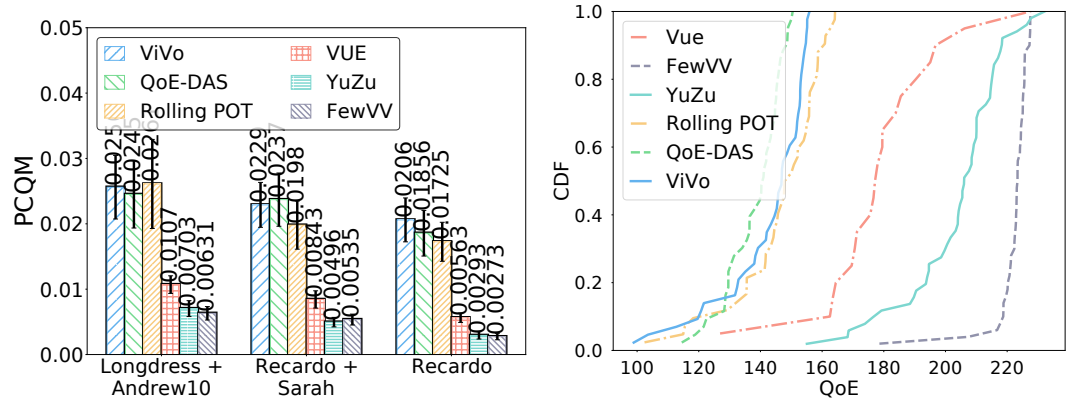
Evaluation Metrics: We use QoE defined in Section. 4.4 as the primary evaluation metric. We adopt two full-reference 3D video quality assessment (VQA) metrics PCQM [36] and point-to-plain PSNR [59] to compute the per tile quality in Eq. 4.1. Because PCQM is a measure of distance, so a smaller PCQM represents a higher quality, while for p2plain-PSNR, a larger PSNR shows higher quality, we use residual of PCQM to balance this difference. The QoE hyper-parameters are determined by the user preference (delay preferred or bandwidth preferred).

We evaluate GVVS (FewVV) with three baseline schemes:



(a) Comparison under Different Sequences (QoE)

(b) Comparison under Different Sequences (PSNR)



(c) Comparison under Different Sequences (PCQM)

(d) Overall QoE CDF on Full Dataset Corpus

Figure 4.5: Overall QoE Performance

- **ViVo [12]**: Vivo uses a decoupled design, it propose a MLP model to predict the viewport, then develop an optimization algorithm to assign proper weights to different tiles to achieve the viewport-oriented tile pruning and optimization, they use a lightweight bitrate estimator to finish the bitrate selection. Because ViVo didn't provide a source code, we reproduce ViVo according to their description.

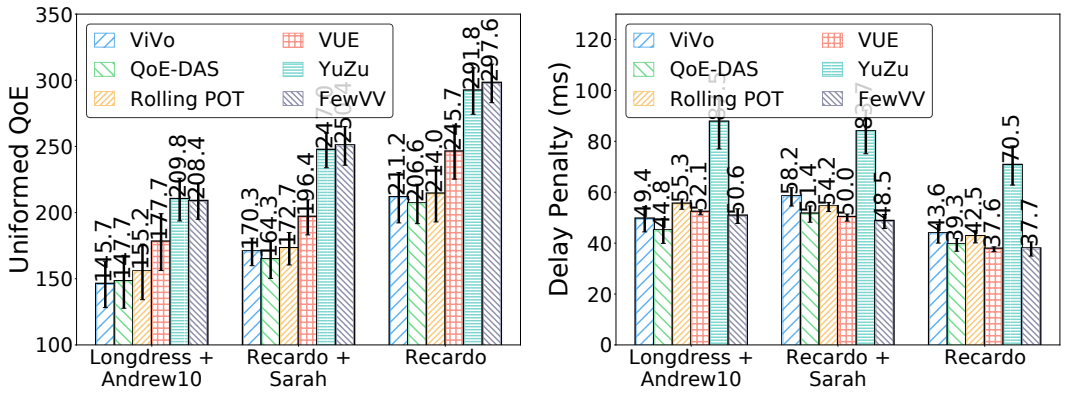
- **Rolling POT [29]**: Rolling POT proposes a prediction-optimization-transmission framework, that achieve a better prediction and optimization coordination.

- **QoE-DAS [63]**: QoE-DAS first propose a innovative QoE model of volumetric video inspired by the perspective projection of the 3D computer graphics rendering process, and then it transforms QoE optimization to a submodular function and proposed a greedy algorithm.

- **Vue [34]**: Vue first studied an edge-assisted transcoding system for volumetric video, they build the first QoE model for mobile edge-assisted volumetric video streaming. It uses a group of small machine learning models to implement a adaptive multiview transcoding scheme, that adapts to bandwidth dynamics and improve the QoE by saving bandwidth consumption.

- **Yuzu [74]**: Yuzu is the first super resolution framework for volumetric video streaming. They optimize the inter and intra frame behavior of the existing 3D super resolution models. Which increase the inference speed of the 3D SR models by 542 times, and allows a real-time 3D super resolution. Their full system implementation shows a good real-world performance at the cost of a slight higher latency penalty.

- **Multi-task Behavior Cloning (MT-BC) [1]**: A offline imitation learning algorithm with meta-learning ability, it is another approach to achieve generalization, we use it in the adaptation ability comparison as baseline. Behavior



(a) Uniform QoE Score

(b) Latency Penalty

Figure 4.6: QoE Components Breakdown.

Cloning learns from experts' trajectories and conducts pairs of states and corresponding optimal actions, it is less general, and more sensitive to training data quality.

4.7.2 Overall Performance

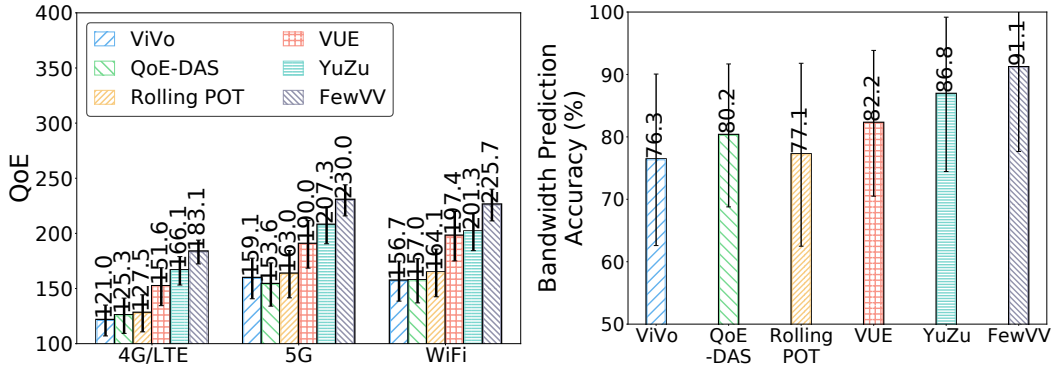
From a perspective of overall QoE, we evaluate the performance of FewVV system.

The QoE Comparison to the existing methods As shown in Fig. 4.5(a), FewVV consistently outperforms the baselines on different video sequences. First, the *Recardo* sequence, since the content of the video remain to be the same person, it requires less generalization ability. FewVV outperforms the Yuzu, Vue, Rolling POT, QoE-DAS and Vivo by 8.66%, 22.85%, 44.68%, 49.17%, and 47.20% respectively. While on the *Longdress+Andrew10* sequence, which starts with *Longdress* sequence in 8i dataset, then followed by *Andrew10* sequence in MVUB. This leads to a distribution shift during a single playback session, and would test the generalization ability of the volumetric video streaming system. Here FewVV outperforms the Rolling ROT, QoE-DAS, and Vivo by 43.60%, 46.12%, and 51.32%. This result shows that FewVV

can generalize to different content features. We also show the point-to-plain PSNR, and PCQM in Fig. 4.5(b) and Fig. 4.5(c), this confirms that our gain is consistent across different visual quality assessment (VQA) metrics. While Fig. 4.5(d) depicts the statistical performance of the FewVV on full video dataset corpus, which shows a consistent higher quality compared to baselines.

QoE Breakdown Analysis The QoE score for each frame is calculated by a linear combination between uniform QoE score and latency penalty, we analyze these two components' impact in Fig 4.6. we can see that the FewVV algorithm's advantage is mainly gained from uniform QoE score (Fig. 4.6(a)), which means FewVV assigns bitrate for each tiling with higher accuracy, and can predict the viewport more accurately, which avoids transmitting the invisible tiles and reduced the bandwidth consumption. On the other hand, we can see in Fig. 4.6(b), FewVV can also get a lower latency penalty, which is 16.64%, 5.64%, and 10.52% percent lower than Vivo, QoE-DAS, and Rolling POT, this is because FewVV has a higher bandwidth prediction accuracy, that prevented using excessive sending rate, thus reduced network queuing delay. This gap widens to 42.07% percent compared to the Yuzu, this is because Yuzu needs to call the 3D super resolution model, which leads to a high model inference delay. Fig. 4.7(b) shows the bandwidth prediction comparison across the baselines, where FewVV demonstrated a higher accuracy, which roots from its better generalization ability and a larger model size, that deliver a superior sequence prediction capability.

The performance under different networks Fig. 4.7(a) evaluates the performance of FewVV under different network environments compared to the baselines. For 4G/LTE networks, FewVV achieves 10.26%, 20.75%, 43.61%, 46.13%, and 51.32% higher QoE respectively compared with Vivo, QoE-DAS, Rolling POT, VUE, Yuzu. For 5G networks, FewVV gains 10.91%, 21.01%, 41.07%, 49.69%, and 44.52%. For



(a) Comparison under Different Network Environment

(b) The Bitrate Prediction Comparison

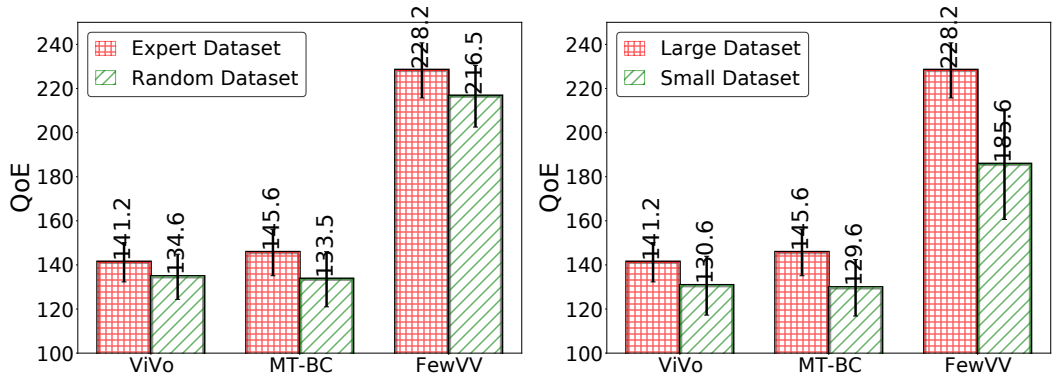
Figure 4.7: The Generalization to the Different Network Condition

WiFi networks, FewVV outperforms other models by 12.15%, 14.32%, 37.51%, 43.76%, and 44.04%. Such results demonstrate the FewVV's strong generalization ability to new network environments.

4.7.3 Ablation Study

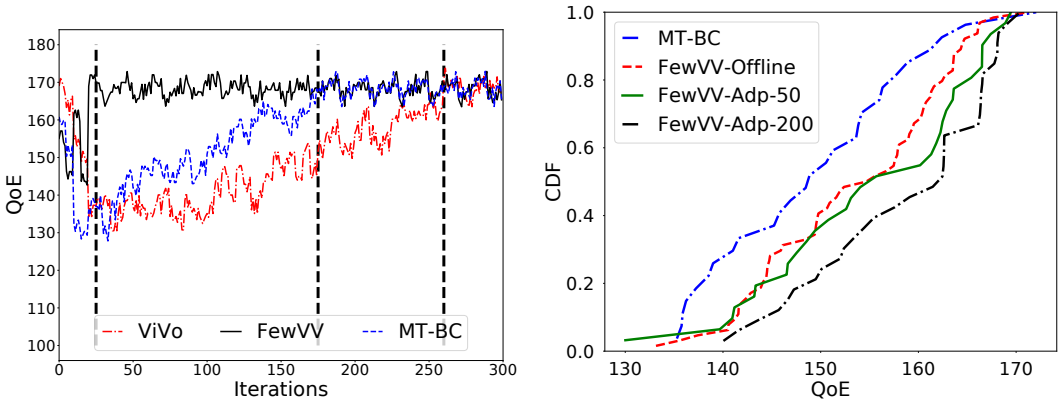
To explore the root cause of the FewVV's gain compared to existing systems. We evaluates the relative importance of different system factors on FewVV.

The impact of the dataset quality and scale Since FewVV is an offline reinforcement learning algorithm, its performance depends on the quality and scale of the training dataset. We evaluate this impact factor here. We first explore the impact of the dataset *quality* in Fig. 4.8(a), we can see the gap between the expert and random datasets are 4.89%, 9.06%, and 5.40% for FewVV, MT-BC, and Vivo respectively. FewVV has a smaller gap than MT-BC, this is because FewVV can learn the intrinsic relationship between the bitrate, viewport, and QoE from random trajectories, it is less sensitive to the training trace quality.



(a) Impact of the pre-training dataset quality (b) Impact of the pre-training dataset size

Figure 4.8: Ablation Study on pre-training dataset characters.



(a) Convergence Speed (b) CDF of Adaptation QoE

Figure 4.9: Ablation Study on Adaptation Speed.

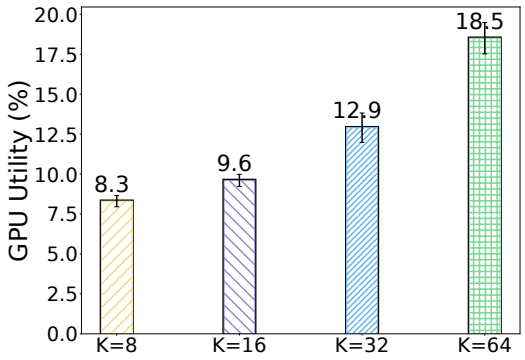


Figure 4.10: Computation Resources

Second, we explore the impact of the dataset scale in Fig. 4.8(b). We can see that the gap between the small and large datasets are 8.13%, 12.33%, and 22.95% respectively for ViVo, MT-BC and FewVV. This is because the FewVV is based on a large language model, it has a slower convergence during the pre-training stage, so it takes more offline data trajectory to train. however, its has a higher asymptotic performance when converged. Although FewVV requires more data in pre-training, the pre-training is done offline, so it has no impact on the run-time adaptation performance of FewVV, we will explore FewVV's run-time adaptation performance in next section.

Fast Adaptation to New Scene and Network Condition In Fig. 4.9(a), we change the sequence from *Andrew10* to *David10* during the run-time, we can see that FewVV converges within 30 iterations, while MT-BC costs 175 iterations and ViVo costs 255 iterations. We also compare the performance of FewVV for offline mode, adaption for 50 iterations, and adaption for 200 iterations respectively.

4.7.4 System Overhead

To explore the real-world applicability of FewVV, we conduct a group of experiments to analyze its system overhead, including analysis on computation resources, mem-

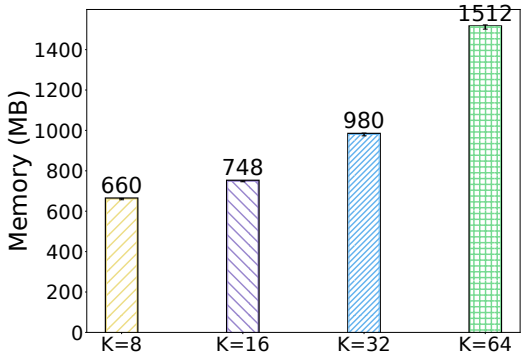


Figure 4.11: Memory Usage

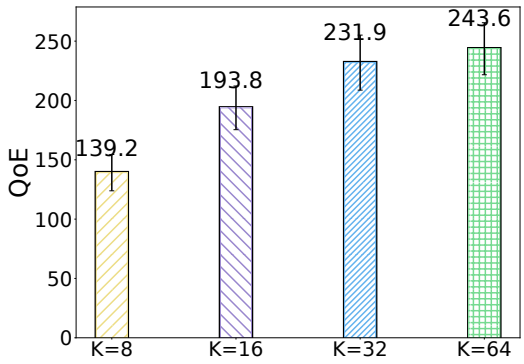


Figure 4.12: Overall System Performance

ory usage, and overall system performance under various conditions. The primary impacting factor here is K, the length of the context window.

Fig. 4.10 depicts the *GPU Utility* of FewVV under different context lengths K. When K increases from 8 to 64, the GPU Utility of FewVV goes up from 8.3% to 18.5%. This is because a longer context length requires more operations within each layer of the VCT model, which naturally increases the computation resource consumption. Similarly, the *Memory Usage* also goes up from 660MB to 1512MB, as shown in Fig. 4.11. Changing K is an effective trade-off between the system overhead and the performance. In Fig. 4.12, we can see how the *Overall system performance* changes according to the K. We observe that when K reaches a sufficient length of 32, the system performance gain will diminish. To sum up, a typical value of K=32 could reach an ideal performance for FewVV, which consumes a GPU utility of 12.9% and a memory usage of 980MB, which is not a bottleneck for most real-world deployment environments. However, a user could choose a proper K that fits into their system capacity in real-world deployment.

4.7.5 Discussion

System Complexity

System complexity is a potential limitation of FewVV, as shown in Sec. 4.7.4. FewVV has a higher system complexity regarding GPU utilization and memory usage. It is mainly because of a larger control model VCT and a longer context window size. Although a larger model and a longer context window size bring a better generalization ability, its complexity is also higher. Nevertheless, in practice, we observe that the system complexity incurred by FewVV is not a bottleneck to the system performance under a typical setting. However, to allow our system to be applicable to a wider range of real-world deployment environments, we think reducing the model complexity is an important direction for future work.

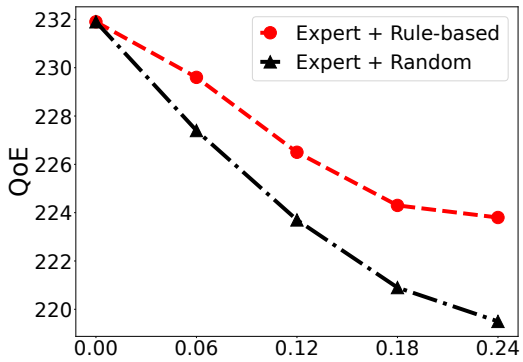


Figure 4.13: Dependency on Dataset Quality

Dependency on Dataset Quality

The dependency of dataset quality is another potential limitation of FewVV. In essence, FewVV will need to learn from the offline collected trajectories. Although it can selectively learn the dataset by recognizing the high reward trajectories, learning a good action from a poor dataset is still challenging. We construct two categories of poor-quality datasets to explore the potential impact of the dataset quality on the FewVV. In the first dataset, we mix the expert (high-quality) trajectories with the rule-based (low-quality) trajectories. By increasing the ratio of the rule-based dataset, we get a training dataset with worse quality. Similarly, we construct the second dataset by mixing expert and random trajectories.

The results are shown in Fig. 4.13. As the quality of the dataset drops, the performance of FewVV drops slightly but is maintained at a high level. We think the quality of the dataset does have an impact on FewVV. Yet, as long as the amount of low-quality data is not dominant, our algorithm can resist its negative effect without losing much performance. But, in future work, it is possible to improve the robustness of FewVV to further reduce the negative impact of poor quality trajectories, allowing the FewVV to be more practical in real-world applications.

4.8 Conclusion

In this article, we studied the generalization problem of the volumetric video streaming system, a type of virtual reality video. The existing systems use DRL-driven viewport-based pruning and bitrate allocation to improve communication efficiency. We examine the limitation when these systems experience a distribution shift, where the system is tested on an unseen dataset with an out-of-distribution environment. To alleviate these limitations, we propose a new framework, GVVS, that allows a few-shot adaptation. We develop a new formulation suitable for sequence predictors, then solve it with a few-shot transformer model. The evaluation demonstrates a significant improvement over the existing system in both in-distribution and out-of-distribution scenarios.

Chapter 5

A Bitstream-corrupted Volumetric Video Dataset for Partial Reliable Error Concealing

5.1 Introduction

As 3D capturing devices like RGB-D camera and LiDAR become more compact and affordable. The 3D media that can support free-viewport experiences is getting more and more attention. Applications such as teleconference, sports game broadcasting, and Metaverse are widely used in commercial activities and educational scenarios for professional and entertainment purposes. Which has now becomes a one of the most dominant applications [44] for the VR [13] and XR [60] devices. There are several formats of the immersive videos, first the 360-degree video, which is the first widely commercialized immersive video format, 360-degree video provides a parametric experience that allows the users to experience a real-world capture environment with excellent quality, it is a killer application for most of the VR headsets like Oculus Quest II, Hololens II, and the Apple Vision Pro. Such video is essentially a projected

video of multiple cameras calibrated with a single origin. Because its compress format is large similar to the existing 2D video frames, it can rides over the current video streaming infrastructure including the Content-Distributing-Network (CDN) for video-on-demand service, and the overlay networks for video conference and Live-streaming services. However, this format cannot provides six-degree-of-freedom experience, that is it only allows the users to rotate the viewport while not changing the viewing position of the viewer, that largely limited its immersiveness for the user. Therefore, several free-viewport video formats are proposed, e.g., volumetric video, Neural Radiance Fields (NeRF) [37], and the 3D Gaussian Splatting [23].

NeRF has the highest quality compared to the volumetric video and 3D Gaussian Splatting, that uses a Multi-Layer-Perception Network to learn an implicit representation of a 3D scene, that given the position and the angle, it outputs directly a radiance level with several parameters. Such technique provides a camera level quality by providing a very accurate lighting fidelity. However, its large amount of parameters cause a very large storage costs, a single frame could have tons' of Gigabytes, making it unsuitable for dynamic scene storage and streaming, even worse, its implicit representation nature making it hard to decompose the frame into blocks, preventing it from being suitable for the viewport-dependant bandwidth saving schemes. As a result, NeRF is not ready for streaming 3D videos.

Therefore, the Volumetric video streaming [12, 74, 34], which stores the 3D content in a collection of points with temporal movements, achieves a perfect balance between the quality and the backward compatibility to the existing Internet architectures, and becomes the dominant video format for the large scale adoption. The raw dynamic point clouds have a high demand for network bandwidth, ranging from 200Mbps to 600Mbps. For today's network devices, such a high throughput could be difficult to transmit over the Internet, involving multiple hops of routers and networks. Therefore, compression is a necessary technique to enable commercial DPC-based applications.

In traditional real-time video streaming, because of the low latency requirement, RTP is widely used as the transport protocol with strong compatibility with the video codecs and relies on the UDP protocol, giving a low communication latency. However, this low latency leads to a break in reliability. Although the video codecs have some tolerance for packet loss, either through FEC redundancy encoding or error concealing methods, it would still cause some damage to the image. This effect is rarely studied in 3D scenarios. Although extensive research is on deep learning-based 2D video restore, there is limited work on the dynamic point cloud. This paper aims to build a bitstream-corrupted volumetric video dataset that reflects a real-world corruption situation.

This paper is organized as follows. First, we propose a corruption model based on the possible patterns of the network loss situations. Second, we build a comprehensive dataset that reflects most of the corruption scenarios. Third, we conduct a detailed analysis of the factors and features of the corruption patterns. Finally, based our observation, we point out the guidelines for designing an ideal learning-based error concealing model for volumetric video based on these discoveries.

We made several original contributions:

- We propose the first error corruption model for the real-time and live-streaming volumetric video. We build a new corruption model that has several new parameters based on real-world network loss models, we argue that our model has a more realistic effect than the existing network agnostic datasets. And therefore providing a more accurate benchmark for the existing error concealing methods on volumetric video streams.

- We open-source one of the first bit-error corrupted volumetric video streaming dataset, that reflects the loss corruptions occurs in the real-time and live streaming scenarios. Where the network are unreliable or partial-reliable due to the strict latency requirements.

- Based on our dataset, we analyzes the corruption performance on different type of volumetric compression codecs. Finally, we analyze the artifact and the implications of these video codecs.

5.2 Dataset

Volumetric Video is a new type of media format that provides 6 DoF experiences to the users. Each frame of a volumetric video is a dense colored point cloud. Which could either be captured by using a single RGB-D camera, or multiple stereo cameras, achieving different level of quality and fidelity's. According to different quality, capture devices, and subject of capture, the volumetric video datasets could be classified to several classes, including the full body dataset 8i, the upper body dataset MVUB, the full scene dataset FSVVD, and so on. These datasets are in its raw format ply, that is a binary encoded points, each points has several parameters, including its position in the space, the color, and lighting information. However, the dense point cloud is a highly bandwidth intensive, there dataset in its raw format is not suitable for Internet video streaming, a typical volumetric video has a bitrate of 1.4Gbps before the compression. Therefore, a range of compression algorithms are proposed to facilitate the video streaming process. MPEG G-PCC and Draco take a Octree-based compression methods, it first organize the points into a Octree, then encodes the position of each points with a incremental coding methods, note that this approach has no temporal compression feature, which means the frames of a sequence are encoded independently.

However, these compression itself cannot provides a adaptive video streaming service, it is necessary to combine them with the specific video streaming service frameworks. DASH and RTP are two representative technologies, whereas DASH is good on its scalability; RTP has a lower latency, and is more responsive. The early stage of the DASH technologies only provides video-on-demand services of long video, like

movie and TV programs. Which has now been replaced by the short videos and live castings. The live streaming services does not use a reliable network protocol, therefore leading to parts of packet loss and bit-errors from times to time. The error concealing technologies is thus proposed to protect and recover the video codecs. Recently, the deep learning-based error concealing of the volumetric video is also proposed for this purpose.

These learning-based error concealing algorithms need a dataset that includes the volumetric videos before and after the corruption and there lacks a benchmark to compare different volumetric error concealing methods. Therefore, in this work we propose the first bitstream-corrupted volumetric video dataset.

Our dataset includes three type of scenes:

- Biterror-corrupted Voxelized Human Body (based on 8i dataset[25])
- Packet-corrupted Upper Body (based on Microsoft Voxelized Upper Body dataset[4])
- Stochastic-corrupted Full Scene Volumetric Video Dataset (based on FSVVD datsaet[14])

For each scene, we corrupt it with different corruption parameters, we provide 16 combinations of the corruption parameters and in total, we have 1.5k frames of corrupted video datasets with its original frame as reference frame, and the corrupted frame with different corruption parameters.

5.3 Corruption Model

Network loss models play a pivotal role in the realm of wireless communication, serving as the cornerstone for accurately depicting the behavior of packet loss in simulated network environments. These models are essential for conducting network simulations

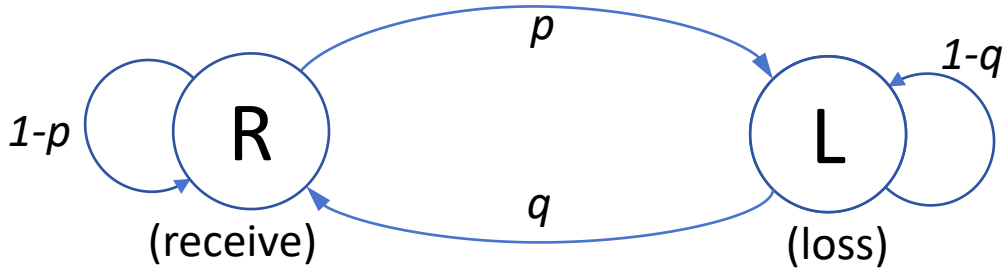


Figure 5.1: Gilbert Model

and emulations, which in turn are critical for predicting system performance, guiding the design of communication protocols, and evaluating the efficacy of various network strategies. The significance of network loss models lies in their ability to encapsulate the inherent unpredictability and complexity of wireless channels, where factors such as multipath fading, shadowing, and interference contribute to the variability in packet loss patterns.

The development of network loss models reflects a transition from basic to complex representations, enhancing the accuracy of wireless network simulations. Initial models like the Bernoulli Model assumed independent packet losses, while subsequent models such as the Gilbert and Gilbert-Elliot Models introduced state transitions to account for loss bursts. Further complexity was added with models like the Three-state Markov Model and the Extended Gilbert Model, which consider multiple states for varied packet loss scenarios.

These models are essential for predicting the quality of video transmission and for devising strategies to mitigate the effects of data loss. As we embark on this exploration, we will methodically review these models, detailing their unique characteristics and applicability to the realm of video streaming.

Bernoulli Model: The Bernoulli Model [42, 2, 51, 67] is a simplistic approach that treats each packet transmission as an independent event with a binary outcome: either a success or a loss. This model assumes that the probability of packet loss remains constant across transmissions and does not account for the bursty nature of packet

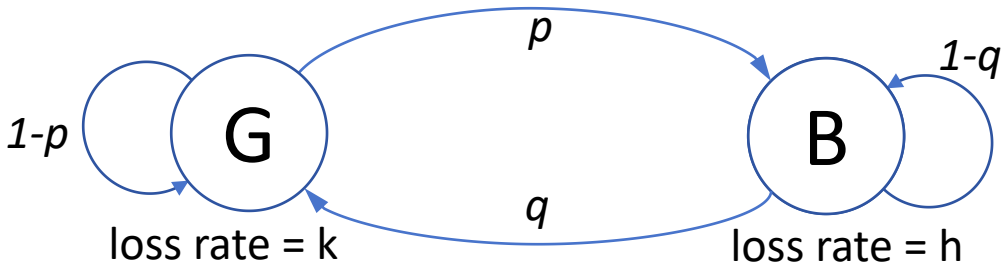


Figure 5.2: Gilbert-Elliot Model

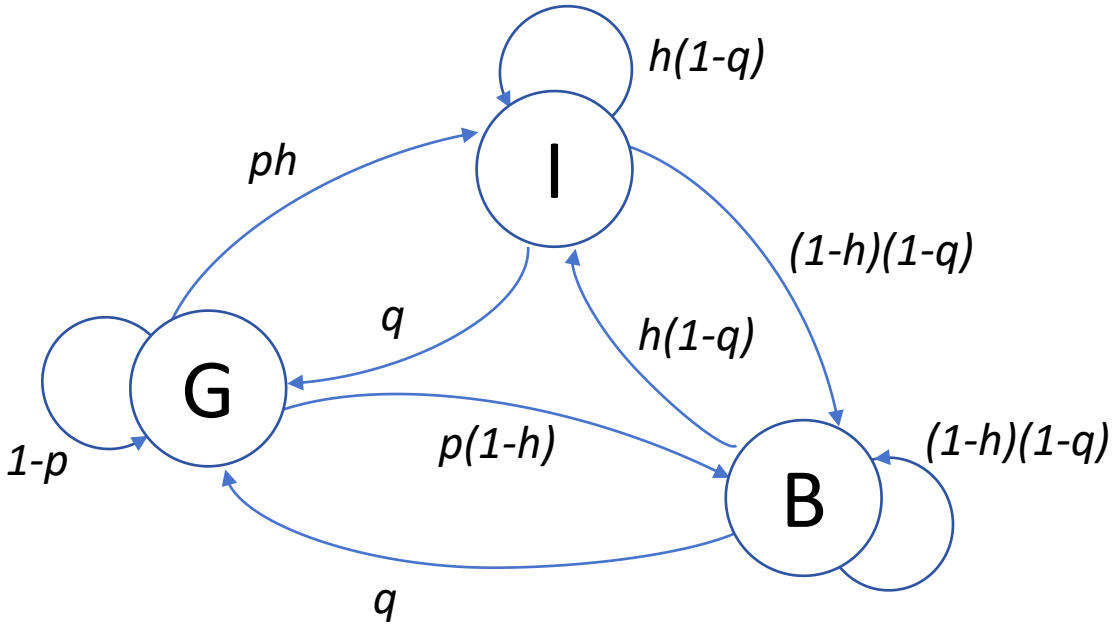


Figure 5.3: Three-state Markov Model

losses. On contrast, Bernoulli model define RRL (reception run-lenth) and LRL (loss run-lenth) as i.i.d variables, where in reception status packets are intact, in contrast, in loss status, the packets would experience a loss. The random variable X_i is set to 1 if the packet i is lost. We define the average loss rate as $\hat{r}$, while RRL, LRL distribution is given as:

$$f_{RRL}(i) = \hat{r}(1-\hat{r})^{i-1} \quad \text{for} \quad i = 1, 2, \ldots, \infty$$
$$f_{LRL}(i) = (1-\hat{r})\hat{r}^{i-1} \quad \text{for} \quad i = 1, 2, \ldots, \infty$$

Gilbert Model [72, 20]: In the Gilbert model, a Markov chain with two state R (Receive) and L (Loss) are introduced. The advantage of Gilbert Model in packet loss simulation is it's capability of capturing the dependency between consecutive packet losses, providing a more accurate representation of burst loss behavior.

The Gilbert Model consider the transmission is always start with the first state of R. In state R, the transmission is error free, while state L indicate packet loss. The probability that state R transit to state L and its reverse process are p and q, respectively. Therefore we have following state transition matrix:

$$\mathbf{P} = \begin{array}{c} \\ R \\ L \end{array} \begin{array}{c} \begin{array}{cc} R & L \end{array} \\ \left\| \begin{array}{cc} 1-p & p \\ q & 1-q \end{array} \right\| \end{array}$$

Gilbert-Elliot Model [54]: This Model is an extended version of Gilbert Model. That allows both the good and bad states to have packet loss, therefore, this model offers a more flexible framework to represent different probabilities of loss in each state, thus accommodating varying network conditions.

Three-state Markov Model [38]: This model adds an intermediary (I) state to the Gilbert-Elliot Model, providing a finer granularity in the transition between loss and no loss states. This model is particularly useful for environments where packet

loss is not solely binary but can involve temporary degradation in signal quality.

Typically, both the state G and state I are error-free. However, the state G have a much more higher self-transition possibility than state I, which leading to a long-term of lossless status in state G and a short-term of lossless status in state I. Therefore, with the help of state I, it can better represent the burst packet loss.

$$\mathbf{P} = \begin{array}{c} \\ G \\ B \\ I \end{array} \begin{array}{c} \begin{array}{cccc} G & B & I \end{array} \\ \left\| \begin{array}{ccc} 1-p & p(1-h) & ph \\ q & (1-h)(1-q) & h(1-q) \\ q & (1-h)(1-q) & h(1-q) \end{array} \right\| \end{array}$$

After implementing these aforementioned network loss models, we can proceed to simulate network transmission to obtain various Bitstream-corrupted Volumetric Videos. Because of different robustness among different encoders, not all Bitstream-corrupted files can be successfully decoded into damaged ply files.

5.4 Dataset Analysis

In this section, we analyze the characteristics of the collected volumtric video dataset with bitstream corruption. Which is based on three uncorrected volumetric video dataset 8i[25], MVUB[4], and FSVVD [14].

Geometry Dominant Loss Effects See Fig. 5.4, in this set of frames, most of the corruption happens on the geometry information, and therefore, leading to the distortion or loss of the geometry information. Which can be roughly classified into two categories: i) the root node loss, where the loss happens on the root nodes or low-depth layer of the octree data structure, this types of loss would cause a blank area in a large portion of the volumetric 3D objects. In our example, the skirt area of

Figure 5.4: The Loss on the Root Node of the Octree Data structure(Geometry Loss Dominant)

Figure 5.5: The Loss on the Color encoding parts(Color Loss Dominant)

the dancer is fully lost. Clearly, this type of loss would cause a significant challenge to the spatial interpolation and temporal prediction-based error concealing schemes. ii) the leaf nodes loss, where the loss happens on the leaf nodes or high-depth layer of the octree data stucture, the effect is a large area with Gaussian Noise distributed while spots. Because the most of the neighbor areas are intact, a spatial interpolation error concealing should be sufficient to recover this type of loss.

Color Distortion Dominant Loss Effects See Fig. 5.5, in this set of frames, the color distortion is dominant, we can see that although the overall geometry structure is correct decoded, the color information is distorted. Instead of total loss of the color, these color error is largely a inaccurate color in a clustered localized area, that is because of the MPEG G-PCC has a spatial localization feature. This provides a solid foundation for allowing the geometry information to guide the color error concealing. We utilize this observation into our new error recovery model design. It is also worth noting that most area of the loss is due to the loss of a single color channel, for example, either red, blue, or green channel is lost. This further simplified the recovery design of the error concealing model.

In summary, the randomness of the loss leads to multiple types of corruption and distortions on both color and geometry information of volumetric video dataset, these losses can be categories into various types and having a diversified characteristics that ask for diversified error concealing technology.

5.5 Related Work

BSVC [33] proposed the first 2D video stream corruption dataset and benchmarked it with a bit-stream level corruption model. They first demonstrate the difference between the existing video corruption dataset and their newly proposed method, then give a plug-and-play module that allows a ready-to-go augmentation of the existing

Figure 5.6: The visual quality of various algorithms on four types of loss.

error-concealing models. They propose a three-parameter video corruption model and demonstrate the distribution of the frame corruption given different corruption parameters. In this way, they achieved a more accurate benchmark of the existing error-concealing models. It allows a new group of error-concealing models to be made. Our work differs from that of the other two, as we target volumetric videos instead of 2D videos. Volumetric video in MPEG V-PCC format is based on the 2D codecs, as it utilizes three different streams and a group of patching and interpolation mechanisms, making it different from the 2D video concealing. Applying the 2D video concealing directly on the MPEG V-PCC video stream does not yield optimal recovery results, as 3D reconstruction has a non-linear relationship between the projected streams and the original stream. As for the MPEG G-PCC-based streams, they also have a different feature: corrupting the part of the video will break the lower levels of the octree data structures, therefore leading to a complex process.

There are several works on 3D error concealing, Fig. 5.6 demonstrate their performance on our dataset, which include both empirical and learning-based methods. Huang et.al. [17] proposed an error-concealing method for MPEG V-PCC-based vol-

umetric video streaming. Based on the analysis of the three streams of the V-PCC video stream, they classified the error into seven categories and found unique loss patterns. They design three approaches. 1) the point-to-point interpolation uses point-level temporal consistency and coherence to achieve a good performance(PI). 2) triangular interpolation (TI), which considers the point itself and the neighborhood points, better utilizing the other two streams of information. 3) the cube motion interpolation. Our work first provides a benchmark for these state-of-the-art error concealing algorithms. And we also propose a new error concealing model, that fully utilize the feature of the volumetric video from both color and geometry information, which is the largest difference from the existing geometry only approaches.

5.6 Conclusion

In this paper, we propose new dataset for the error concealing of the volumetric videos. Volumetric video is an important type of video formats that has low computation cost, high quality, and high data volume. It is potentially the future of the VR, AR, and Metaverse applications, e.g., the teleconferencing, remote medical, gaming, and live sports broadcasting. However, to transmit these video streaming over the Internet, the partial reliable or unreliable channels are necessary to get a low latency experiences, which has been widely proofed in the 2D video streaming applications like 2D video conferencing, Live Streaming, and IPTV applications. This is because the reliable transmission protocols have a high latency overhead due to its re transmission mechanisms. Therefore, any real-time or responsiveness video stream is prune to the bit-errors or packet loss in the bitstream, leading to the corruption of the video stream. Such corruptions become a more serious problem due to the intra and inter-frame coding of the video compression algorithms. We propose a bit stream corrupted VVS dataset, that includes a wide range of volumetric videos representing most of the public volumetric video datasets. We build a realistic loss

model to reflect the bit-error and packet loss under the real-world scenarios. Based on this model, we build a large dataset according to various network conditions and video sequence encoded with three mainstream volumetric compression coding. In conclusion, this paper gives a new dataset that simulate the real-world real-time volumetric video streaming corruption scenarios, which provides the foundation for the evaluation and development of the error concealing methods and future error-resilient 3D scene representation formats.

Chapter 6

Conclusions and Future Directions

6.1 Future Directions

There are several potential directions for further improving the volumetric video streaming systems. First, it is worth noting that some more advanced 3D representation technologies have recently been proposed, e.g., 3D Gaussian Splatting (3DGS), which gives a high-fidelity representation of a 3D scene in a photo-realistic quality. However, the storage format of 3DGS is a straightforward extension of volumetric video codecs. Therefore, there is a potential to extend our approach to support the more advanced 3DGS video streaming. Nevertheless, the state-of-the-art 3DGS format is not readily streamable due to the high bandwidth consumption, lack of efficient compressing codecs, and high decoding and rendering costs that impose a large computational overhead on the existing edge devices. Some pioneer works have been on reducing the transmission costs of 3DGS adaptive bitrate control for 3DGS. In our view, there are many challenges to this emerging problem. So, extending our adaptive volumetric video streaming frameworks to support 3DGS and building a rate-distortion model for 3DGS is very promising and meaningful future work to do.

The second is extending the system to be aware of the multiple user scenarios. Our

current system mainly serves a single viewer within a scene. However, in many application scenarios, having multiple viewers sharing a single watching space is common. For example, in a remote medical teaching class, many students would share a single classroom and watch the same scene. In this scenario, the interaction between viewers has two major impacts on the volumetric video streaming process. First is the sharing of the network channels. Because these users typically share a router in a single room. In this case, balancing the available bandwidth for these users could have an impact on the overall quality of experience (QoE). There is a need for a new QoE metric that considers the multiple users aggregated experience and consider fairness. Second is the viewport prediction correlation. The viewport of these users is not independent. They can occlude each other's light-of-sight. These problems are highly suitable for a multi-variate sequence model. Therefore, extending our approach can potentially solve this challenge with more advanced QoE modeling and an inter-dependency-aware multi-variate sequence model. We leave this for future work.

The third is building an error-concealing model to support partial, reliable, real-time volumetric video streaming. Latency requirements are essential for any real-time video streaming system. These stringent latency requirements lead to a design choice of unreliable or partially reliable network protocols. This is because the reliability is based on re-transmission and a large buffer, which largely expands the system's latency. Therefore, it is important to have error-concealing mechanisms to recover the video stream from error. In this direction, we built an early-stage dataset to explore this problem. By benchmarking several existing methods, temporal interpolation, spatial interpolation, and temporal prediction, we observe their limitation for volumetric video streaming. In the future, we think building a deep learning-based error concealing model for volumetric video recovery will be another important work.

6.2 Conclusion

In this thesis, we studied the volumetric video streaming problem. The volumetric video provides a free viewport viewing experience but also burdens the existing Internet infrastructures with its high volume. Its higher level of information also imposes a greater threat to the privacy of content sharing. This thesis first presents the background of Volumetric Video Streaming, the basic architectures, challenges, and opportunities. Then, it provides a comprehensive literature review that covers the existing works in the field of 2D and volumetric video streaming adaptation and quality-of-experience metrics. Then, for the three aspects of volumetric video streaming, performance, generalization, and robustness, we reveal three key research problems: First, how to effectively stream the volumetric video over the complex Internet according to the fast varying network conditions and user's viewing behaviors; Second, how to achieve generalization of the tile pruning-based volumetric video streaming systems; Third, how to transmit the volumetric video over lossy network channels that provides low latency? To address these research problems, we conduct a study and measurement to reveal the potential solution approaches and gain space, and for each of them, we give a solution. This thesis then makes three contributions: First, it proposes a new MPEG V-PCC-based volumetric video streaming framework that supports a backward-compatible HTTP video streaming for volumetric videos, during which it composes a combination of a new rate-distortion model for the V-PCC-based volumetric video rate control, a new offline reinforcement learning-based bitrate adaption algorithm, and a frame rate scaling mechanism to improve the smoothness of the playback by filling the gap between the coarse-grained DASH control and fine-grained network variations. Second, we propose a few-shot adaptation framework to solve the generalization problem in the tile pruning-based volumetric video streaming systems. Third, this thesis proposes a novel bitstream corrupted volumetric video dataset to support partially reliable volumetric video streaming. As the latency demands of volumetric video streaming systems get more stringent,

a shift from a high-latency reliable channel to a low-latency unreliable channel has happened. Therefore, a group of error-concealing methods is proposed to mitigate the error exposed from the network channel to the application decoder. However, a comprehensive and realistic dataset for corrupted volumetric videos is lacking. We fill this gap and propose a network-inspired corruption model. We made an in-depth qualitative discussion of our dataset and pointed out the patterns of the loss-induced frame artifacts, which provide a practical guideline for designing future learning-based error-concealing models for volumetric video. We evaluated our systems and observed a meaningful gain. In conclusion, we discuss future works and summarize this thesis.

References

[1] Alex Beeson and Giovanni Montana. Improving td3-bc: Relaxed policy constraint for offline learning and stable online fine-tuning. *arXiv preprint arXiv:2211.11802*, 2022.

[2] J-C Bolot, Sacha Fosse-Parisis, and Don Towsley. Adaptive fec-based error control for internet telephony. In *IEEE INFOCOM'99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No. 99CH36320)*, volume 3, pages 1453–1460. IEEE, 1999.

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[4] Loop Charles, Cai Qin, et al. Microsoft voxelized upper bodies - a voxelized point cloud dataset.

[5] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision trans-

former: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

[6] Federico Chiariotti, Stefano D'Aronco, Laura Toni, and Pascal Frossard. Online learning adaptation strategy for dash clients. In *Proc. of ACM MMSys'16*, Klagenfurt, Austria, May 2016.

[7] Lucas Clemente and Marten Seemann. Quic-go, 2019.

[8] Alvaro Collet, Ming Chuang, Pat Sweeney, Don Gillett, Dennis Evseev, David Calabrese, Hugues Hoppe, Adam Kirk, and Steve Sullivan. High-quality streamable free-viewpoint video. *ACM Transactions on Graphics*, 34(4):1–13, 2015.

[9] Jens de Hoog, Ahmed N Ahmed, Ali Anwar, Steven Latré, and Peter Hellinckx. Quality-aware compression of point clouds with google draco. In *Procs. of 3PGCIC'21*, pages 227–236. Springer, 2021.

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

[11] John K Haas. A history of the unity game engine. *Diss. Worcester Polytechnic Institute*, 483(2014):484, 2014.

[12] Bo Han, Yu Liu, and Feng Qian. Vivo: Visibility-aware mobile volumetric video streaming. In *ACM MobiCom'20*, 2020.

[13] Cornel Hillmann and Cornel Hillmann. Comparing the gear vr, oculus go, and oculus quest. *Unreal for Mobile and Standalone VR: Create Professional VR Apps Without Coding*, pages 141–167, 2019.

[14] Kaiyuan Hu, Yili Jin, Haowen Yang, Junhua Liu, and Fangxin Wang. Fsvvd: A dataset of full scene volumetric video. In *Proceedings of the 14th Conference on ACM Multimedia Systems*, pages 410–415, 2023.

[15] Te-Yuan Huang, Ramesh Johari, Nick McKeown, Matthew Trunnell, and Mark Watson. A buffer-based approach to rate adaptation: Evidence from a large video streaming service. In *Proc. of ACM SIGCOMM'14*, Chicago, USA, August 2014.

[16] Tianchi Huang, Rui-Xiao Zhang, Chao Zhou, and Lifeng Sun. Qarc: Video quality aware rate control for real-time video streaming based on deep reinforcement learning. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 1208–1216, 2018.

[17] Tzu-Kuan Hung, I-Chun Huang, Samuel Rhys Cox, Wei Tsang Ooi, and Cheng-Hsin Hsu. Error concealment of dynamic 3d point cloud streaming. In *Proceedings of the 30th ACM International Conference on Multimedia*, MM '22, page 3134–3142, New York, NY, USA, 2022. Association for Computing Machinery.

[18] Alireza Javaheri, Catarina Brites, Fernando Pereira, and João Ascenso. Improving psnr-based quality metrics performance for point cloud geometry. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 3438–3442. IEEE, 2020.

[19] Junchen Jiang, Vyas Sekar, and Hui Zhang. Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive. In *Proc. of ACM CoNEXT'12*, Nice, France, December 2012.

[20] Wenyu Jiang and Henning Schulzrinne. Modeling of packet loss and delay and their effect on real-time multimedia service quality. In *Proc. NOSSDAV*, pages 1–10, 2000.

[21] Kaggle. Ookla internet speed dataset, 2022.

[22] Nuowen Kan, Yuankun Jiang, Chenglin Li, Wenrui Dai, Junni Zou, and Hongkai Xiong. Improving generalization for neural adaptive video streaming via meta reinforcement learning. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 3006–3016, 2022.

[23] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4), jul 2023.

[24] Turgay Korkmaz and Marwan Krunz. Multi-constrained optimal path selection. In *Procs' of IEEE INFOCOM'2001*, volume 2, pages 834–843. IEEE, 2001.

[25] Maja Krivokuca, Philip A Chou, and Patrick Savill. 8i voxelized surface light field (8ivslf) dataset. *ISO/IEC JTC1/SC29/WG11 MPEG, input document m42914*, 2018.

[26] Jason Lawrence, Dan B Goldman, Supreeth Achar, Gregory Major Blascovich, Joseph G. Desloge, Tommy Fortes, Eric M. Gomez, Sascha Häberling, Hugues Hoppe, Andy Huibers, Claude Knaus, Brian Kuschak, Ricardo Martin-Brualla, Harris Nover, Andrew Ian Russell, Steven M. Seitz, and Kevin Tong. Project starline: A high-fidelity telepresence system. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 40(6), 2021.

[27] Davi Lazzarotto, Evangelos Alexiou, and Touradj Ebrahimi. Benchmarking of objective quality metrics for point cloud compression. In *IEEE 23rd International Workshop on Multimedia Signal Processing*, pages 1–6. IEEE, 2021.

[28] Kyungjin Lee, Juheon Yi, Youngki Lee, Sunghyun Choi, and Young Min Kim. Groot: a real-time streaming system of high-fidelity volumetric videos. In *ACM MobiCom'20*, 2020.

[29] Jie Li, Huiyu Wang, Zhi Liu, Pengyuan Zhou, Xianfu Chen, Qiyue Li, and Richang Hong. Toward optimal real-time volumetric video streaming: A rolling

optimization and deep reinforcement learning based approach. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(12):7870–7883, 2023.

[30] Jinyang Li, Zhenyu Li, Ri Lu, Kai Xiao, Songlin Li, Jufeng Chen, Jingyu Yang, Chunli Zong, Aiyun Chen, Qinghua Wu, et al. Livenet: a low-latency video transport network for large-scale live streaming. In *Proceedings of the ACM SIGCOMM 2022 Conference*, pages 812–825, 2022.

[31] Linux.org. Netem-network emulator, November 2011.

[32] Junhua Liu, Boxiang Zhu, Fangxin Wang, Yili Jin, Wenyi Zhang, Zihan Xu, and Shuguang Cui. Cav3: Cache-assisted viewport adaptive volumetric video streaming. In *2023 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*, pages 173–183, 2023.

[33] Tianyi Liu, Kejun Wu, Yi Wang, Wenyang Liu, Kim-Hui Yap, and Lap-Pui Chau. Bitstream-corrupted video recovery: A novel benchmark dataset and method. *Advances in Neural Information Processing Systems*, 36, 2024.

[34] Yu Liu, Bo Han, Feng Qian, Arvind Narayanan, and Zhi-Li Zhang. Vues: practical mobile volumetric video streaming through multiview transcoding. In *Procs of the 28th Annual International Conference on Mobile Computing And Networking (MobiCom'22)*, Sydney, NSW, Australia, October 2022. ACM.

[35] Markets and Markets. Volumetric video market by volumetric capture (hardware, software, services), application (sports, events, and entertainment, medical, advertisement, and education), content delivery and region (2021-2026), 2022.

[36] Gabriel Meynet, Yana Nehmé, Julie Digne, and Guillaume Lavoué. Pcqm: A full-reference quality metric for colored 3d point clouds. In *2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX)*, pages 1–6. IEEE, 2020.

[37] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[38] Ben P Milner and Alastair Bruce James. An analysis of packet loss models for distributed speech recognition. In *Eighth International Conference on Spoken Language Processing*, 2004.

[39] MPEG. Mpeg video point cloud compression reference software, 2022.

[40] MPEG-I. Common test conditions for immersive video, 2022.

[41] Elias Neuman-Donihue, Michael Jarvis, and Yuhao Zhu. Fastpoints: A state-of-the-art point cloud renderer for unity. *arXiv preprint arXiv:2302.05002*, 2023.

[42] Dong Nguyen, Tuan Tran, Thinh Nguyen, and Bella Bose. Wireless broadcast using network coding. *IEEE Transactions on Vehicular technology*, 58(2):914–925, 2008.

[43] Sean Ong, Varun Kumar Siddaraju, Sean Ong, and Varun Kumar Siddaraju. Introduction to the mixed reality toolkit. *Beginning Windows Mixed Reality Programming: For HoloLens and Mixed Reality Headsets*, pages 85–110, 2021.

[44] Sergio Orts-Escolano, Christoph Rhemann, Sean Fanello, Wayne Chang, Adarsh Kowdle, Yury Degtyarev, David Kim, Philip L Davidson, Sameh Khamis, Mingsong Dou, et al. Holoportation: Virtual 3d teleportation in real-time. In *Procs. of the 29th annual symposium on user interface software and technology*, pages 741–754, 2016.

[45] Sida Peng, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Representing volumetric videos as dynamic mlp maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4252–4262, 2023.

[46] Shiva Raj Pokhrel and Michel Mandjes. Internet of drones: Improving multi-path tcp over wifi with federated multi-armed bandits for limitless connectivity. *Drones*, 7(1):30, 2022.

[47] Maurice Quach, Aladine Chetouani, Giuseppe Valenzise, and Frédéric Dufaux. A deep perceptual metric for 3d point clouds. *Electronic Imaging*, 2021(9):257–1, 2021.

[48] Darijo Raca, Dylan Leahy, Cormac J. Sreenan, and Jason J. Quinlan. Beyond throughput, the next generation: A 5g dataset with channel and context metrics. In *Procs. of ACM MMSys'20*, MMSys '20, page 303–308, 2020.

[49] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[50] Michael Rudow, Francis Y Yan, Abhishek Kumar, Ganesh Ananthanarayanan, Martin Ellis, and KV Rashmi. Tambur: Efficient loss recovery for videoconferencing via streaming codes. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, pages 953–971, 2023.

[51] Henning A Sanneck and Georg Carle. Framework model for packet loss metrics based on loss runlengths. In *Multimedia Computing and Networking 2000*, volume 3969, pages 177–187. SPIE, 1999.

[52] Oliver Schreer, Ingo Feldmann, Peter Kauff, Peter Eisert, Danny Tatzelt, Cornelius Hellge, Karsten Müller, Sven Bliedung, and Thomas Ebner. Lessons learned during one year of commercial volumetric video production. *SMPTE Motion Imaging Journal*, 129(9):31–37, 2020.

[53] Yuang Shi, Pranav Venkatram, Yifan Ding, and Wei Tsang Ooi. Enabling low bit-rate mpeg v-pcc-encoded volumetric video streaming with 3d sub-sampling.

In *Procs of the 14th Conference on ACM Multimedia Systems(MMSys '23)*, New York, NY, USA, 2023. Association for Computing Machinery.

[54] Carlos Alexandre Gouvea Da Silva and Carlos Marcelo Pedroso. Mac-layer packet loss models for wi-fi networks: A survey. *IEEE Access*, 7:180512–180531, 2019.

[55] Ashish Singhadia, Preeti Samhita Pati, Chetna Singhal, and Indrajit Chakrabarti. Efficient hevc encoding to meet bitrate and psnr requirements using parametric modeling. *Circuits, Systems, and Signal Processing*, pages 1–33, 2022.

[56] Kevin Spiteri, Rahul Urgaonkar, and Ramesh K Sitaraman. Bola: Near-optimal bitrate adaptation for online videos. *IEEE/ACM Transactions on Networking*, 28(4):1698–1711, 2020.

[57] Liyang Sun, Tongyu Zong, Siquan Wang, Yong Liu, and Yao Wang. Tightrope walking in low-latency live streaming: Optimal joint adaptation of video rate and playback speed. In *Proceedings of the 12th ACM Multimedia Systems Conference*, pages 200–213, 2021.

[58] Yi Sun, Xiaoqi Yin, Junchen Jiang, Vyas Sekar, Fuyuan Lin, Nanshu Wang, Tao Liu, and Bruno Sinopoli. Cs2p: Improving video bitrate selection and adaptation with data-driven throughput prediction. In *Proc. of ACM SIGCOMM'16*, New York, NY, USA, August 2016.

[59] Dong Tian, Hideaki Ochimizu, Chen Feng, Robert Cohen, and Anthony Vetro. Geometric distortion metrics for point cloud compression. In *2017 IEEE International Conference on Image Processing (ICIP'17)*, pages 3460–3464, Beijing, China, September 2017. IEEE.

[60] Dorin Ungureanu, Federica Bogo, Silvano Galliani, Pooja Sama, Xin Duan, Casey Meekhof, Jan Stühmer, Thomas J Cashman, Bugra Tekin, Johannes L

Schönberger, et al. Hololens 2 research mode as a tool for computer vision research. *arXiv preprint arXiv:2008.11239*, 2020.

[61] J. vanderHooft. 4g/lte bandwidth logs, November 2011.

[62] Bo Wang, Yuan Zhang, Size Qian, Zipeng Pan, and Yuhong Xie. A hybrid receiver-side congestion control scheme for web real-time communication. In *Proceedings of the 12th ACM Multimedia Systems Conference*, pages 332–338, 2021.

[63] Lisha Wang, Chenglin Li, Wenrui Dai, Shaohui Li, Junni Zou, and Hongkai Xiong. Qoe-driven adaptive streaming for point clouds. *IEEE Transactions on Multimedia*, 25:2543–2558, 2023.

[64] Ziyi Wang, Yong Cui, Xiaoyu Hu, Xin Wang, Wei Tsang Ooi, Zhen Cao, and Yi Li. Multilive: Adaptive bitrate control for low-delay multi-party interactive live streaming. *IEEE/ACM Transactions on Networking*, 30(2):923–938, 2021.

[65] Jian Xiong, Hao Gao, Miaohui Wang, Hongliang Li, and Weisi Lin. Occupancy map guided fast video-based dynamic point cloud coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(2):813–825, 2021.

[66] Mengdi Xu, Yuchen Lu, Yikang Shen, Shun Zhang, Ding Zhao, and Chuang Gan. Hyper-decision transformer for efficient online policy adaptation. *arXiv preprint arXiv:2304.08487*, 2023.

[67] Maya Yajnik, Sue Moon, Jim Kurose, and Don Towsley. Measurement and modelling of the temporal dependence in packet loss. In *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*. IEEE, 1999.

[68] Qi Yang, Yipeng Liu, Siheng Chen, Yiling Xu, and Jun Sun. No-reference point cloud quality assessment via domain adaptation. In *Procs. of the IEEE/CVF CVPR'22*, pages 21179–21188, 2022.

[69] Xu Yi, Lu Yao, and Wen Ziyu. Owlii dynamic human mesh sequence dataset. In *120th MPEG Meeting ISO/IEC JTC1/SC29/WG11 m41658*, Macau, Oct. 2017.

[70] Xiaoqi Yin, Abhishek Jindal, Vyas Sekar, and Bruno Sinopoli. A control-theoretic approach for dynamic adaptive video streaming over http. In *ACM SIGCOMM'15*, London, UK, August 2015.

[71] Gareth W Young, Néill O'Dwyer, and Aljosa Smolic. A virtual reality volumetric music video: featuring new pagans. In *Proceedings of the 13th ACM Multimedia Systems Conference*, pages 331–333, 2022.

[72] Xunqi Yu, James W Modestino, and Xusheng Tian. The accuracy of gilbert models in predicting packet-loss statistics for a single-multiplexer network model. In *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, volume 4, pages 2602–2612. IEEE, 2005.

[73] Emin Zerman, Néill O'Dwyer, Gareth W Young, and Aljosa Smolic. A case study on the use of volumetric video in augmented reality for cultural heritage. In *Proceedings of the 11th Nordic Conference on Human-Computer Interaction: Shaping Experiences, Shaping Society*, pages 1–5, 2020.

[74] Anlan Zhang, Chendong Wang, Bo Han, and Feng Qian. YuZu: Neural-Enhanced volumetric video streaming. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, pages 137–154, Renton, WA, April 2022. USENIX Association.

[75] Ding Zhang, Bo Han, Parth Pathak, and Haoliang Wang. Innovating multi-user volumetric video streaming through cross-layer design. In *Proceedings of the Twentieth ACM Workshop on Hot Topics in Networks*, pages 16–22, 2021.

[76] Ding Zhang, Puqi Zhou, Bo Han, and Parth Pathak. M5: Facilitating multi-user volumetric content delivery with multi-lobe multicast over mmwave. In *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*,

SenSys '22, page 31–46, New York, NY, USA, 2023. Association for Computing Machinery.

[77] Huanhuan Zhang, Anfu Zhou, Yuhan Hu, Chaoyue Li, Guangping Wang, Xinyu Zhang, Huadong Ma, Leilei Wu, Aiyun Chen, and Changhui Wu. Loki: improving long tail performance of learning-based real-time video adaptation by fusing rule-based models. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, pages 775–788, 2021.

[78] Huanhuan Zhang, Anfu Zhou, Jiamin Lu, Ruoxuan Ma, Yuhan Hu, Cong Li, Xinyu Zhang, Huadong Ma, and Xiaojiang Chen. Onrl: improving mobile video telephony via online reinforcement learning. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, pages 1–14, 2020.

[79] Yuanxing Zhang, Pengyu Zhao, Kaigui Bian, Yunxin Liu, Lingyang Song, and Xiaoming Li. Drl360: 360-degree video streaming with deep reinforcement learning. In *Procs. of the 38th International Conference on Computer Communications (INFOCOM'19)*, Paris, France, June. 2019.

[80] Anfu Zhou, Huanhuan Zhang, Guangyuan Su, Leilei Wu, Ruoxuan Ma, Zhen Meng, Xinyu Zhang, Xiufeng Xie, Huadong Ma, and Xiaojiang Chen. Learning to coordinate video codec with transport protocol for mobile video telephony. In *The 25th Annual International Conference on Mobile Computing and Networking*, pages 1–16, 2019.

[81] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018.