UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

ADAPTIVE SCALABLE VIDEO STREAMING BASED ON DEADLINE-SENSITIVE CRITERIA

TESIS PARA OPTAR AL GRADO DE DOCTOR EN INGENIERÍA ELÉCTRICA

ELIECER RAMON PEÑA ANCAVIL

Profesor Guía:
Claudio Estévez Montero

MIEMBROS DE LA COMISIÓN:
Néstor Becerra Yoma
Shaharyar Kamal
Xianfeng Tang
Fernando Huenupan Quinan

SANTIAGO DE CHILE
2024

RESUMEN DE LA TESIS PARA OPTAR
AL GRADO DE DOCTOR EN
INGENIERÍA ELÉCTRICA
POR: **ELIECER RAMON PEÑA ANCAVIL**
FECHA: 2024
PROF. GUÍA: CLAUDIO ESTÉVEZ MONTERO

TRANSMISIÓN DE VIDEO ESCALABLE ADAPTATIVA BASADA EN CRITERIOS SENSIBLES A FECHAS LÍMITE

El streaming de video domina el uso de datos en Internet, pero entregar una calidad constante es complicado. El método Dynamic Adaptive Streaming over HTTP (DASH), a pesar de su popularidad, sufre interrupciones por búferes vacíos. Para mejorar esto, se propone el protocolo Adaptive Scalable Video Streaming (ASViS), que utiliza codificación de video escalable y un enfoque User Datagram Protocol (UDP), priorizando la entrega puntual. ASViS se adapta a la red como Transmission Control Protocol (TCP), pero es innovador al descartar datos basándose en su relevancia temporal para evitar información obsoleta, cumpliendo con el Request for Comments (RFC) 8085 y previniendo la saturación de la red. Se desarrolló un modelo para prever su comportamiento en distintas condiciones de red y evaluar la influencia de varios parámetros en su rendimiento. La ventaja de ASViS radica en maximizar la calidad de imagen ajustándose a la Scalable Video Coding (SVC) y a las capas de datos disponibles, superando así las limitaciones de DASH y apuntando a un servicio de streaming más fiable y de alta calidad. Esta innovación promete un significativo avance para el streaming de video, buscando mejorar la experiencia del usuario.

THESIS ABSTRACT FOR THE
DEGREE OF DOCTOR IN
ELECTRICAL ENGINEERING
BY: **ELIECER RAMON PEÑA ANCAVIL**
DATE: 2024
THESIS ADVISOR: CLAUDIO ESTÉVEZ MONTERO

ADAPTIVE SCALABLE VIDEO STREAMING BASED ON DEADLINE-SENSITIVE CRITERIA

Multimedia video streaming, identified as the dominant internet data consumption service, brings forth challenges in consistently delivering optimal video quality. Dynamic Adaptive Streaming over HTTP (DASH), while prevalent, often encounters buffering problems, causing video pauses due to empty video buffers. This study introduces the Adaptive Scalable Video Streaming (ASViS) protocol as a solution. ASViS incorporates scalable video coding, a flow-controlled User Datagram Protocol (UDP), and deadline-based criteria. A model is developed to predict the behavior of ASViS across varying network conditions. Additionally, the effects of diverse parameters on ASViS performance are evaluated. ASViS adjusts data flow similarly to the Transmission Control Protocol (TCP), based on bandwidth availability. Data are designed to be discarded by ASViS according to video frame deadlines, preventing outdated information transmission. Compliance with RFC 8085 ensures the internet is not overwhelmed. With its scalability feature, ASViS achieves the highest possible image quality per frame, aligning with Scalable Video Coding (SVC) and the available data layers. The introduction of ASViS offers a promising approach to address the challenges faced by DASH, potentially providing more consistent and higher-quality video streaming.

Agradecimientos

En 2007, al comenzar a estudiar ingeniería, pensé: "Espero salir luego de la universidad". Hoy, siendo ya 2024, sigo vinculado a distintas universidades, como docente, funcionario y, por supuesto, como alumno. Mirando atrás, quizás era un poco inocente, o al menos mis planes tomaron un gran y agradable giro.

Quisiera utilizar este espacio para agradecer a mi familia, que ha sido un pilar fundamental durante todo este proceso, apoyándome, escuchándome y siendo más sabios que yo en saber cuándo hacer una pausa para cuidar de mi salud mental.

A lo largo de mi vida como alumno he tenido muchos profesores, y desafortunadamente el espacio no me permite nombrar a todos los excelentes docentes que me guiaron, me inspiraron y me proporcionaron las herramientas que hoy me convierten en investigador. Sin embargo, me gustaría hacer una excepción y mencionar a mi madre, una mente brillante y disciplinada, que sabe que las recompensas se consiguen con esfuerzo.

También quisiera agradecer especialmente a mi tutor y amigo Claudio Estévez, quien tuvo la paciencia de guiarme y apoyarme tanto moral, disciplinaria como operativamente. Sin él no hubiera logrado llegar hasta aquí hoy; hizo mucho más de lo que se espera de un tutor o incluso de un docente, y las páginas disponibles no me alcanzan para agradecerle lo suficiente.

Y finalmente, quisiera aprovechar también para agradecer al comité de postgrado, que con sus amables y constantes recordatorios sobre la urgencia de ciertos procesos, junto con su gestión, ayudaron a reencausar los esfuerzos para poder llegar hasta aquí.

Table of Content

Table index

Illustrations Index

Acronyms List

AAHCC: Adaptation-Aware Hybrid Client-Cache.
ACK: Acknowledgments.
ABMA+: Adaptation and Buffer Management Algorithm.
ASViS: Adaptive Scalable Video Streaming.
AIMD: Additive Increase/Multiplicative Decrease.
AVC: Advanced Video Coding.
AI: Artificial Intelligence.
AVQ: Average Video Quality.
AVQV: Average Video Quality Variation.
BL: Base Layer.
BL-B: Base Layer Bi-predicted.
BL-I: Base Layer Interpredicted.
BL-P: Base Layer Predicted.
B-frames: Bi-predicted frames.
BOLA: Buffer Occupancy based Lyapunov Algorithm.
BTV: Buffer Threshold Values.
BM: Buffer Map.
CPU: Central Processing Unit.
CTU: Coding Tree Unit.
CU: Coding Units.
CA: Congestion Avoidance.
cwnd: Congestion Window.
CARA: Content-aware Rate Adaptation Algorithm.
CABAC: Context Adaptive Binary Arithmetic Coding.
CAVLC: Context-Adaptive Variable Length Coding.
CS2P: Cross Session Stateful Predictor.
DSCA: Deadline-sensitive Criteria Algorithm.
DL: Dependency Level.
DLM: Detail Loss Metric.
DCT: Discrete Cosine Transform.
DFT: Discrete Fourier Transform.
DASH: Dynamic Adaptive Streaming over HTTP.
EL: Enhancement Layer.
EL-B: Enhancement Layer Bi-predicted.
EL-I: Enhancement Layer Interpredicted.
EL-P: Enhancement Layer Predicted.
EPS: Estimated Packet Size.
XML: Extensible Markup Language.
FESTIVE: Fair, Efficient, and Stable Adaptive.
FRC: Fast Recovery.
FRT: Fast Retransmit.
UHD: Full Ultra High Definition.

GOP: Group Of Pictures.
HEVC: High Efficiency Video Coding.
I-frames: Intra-frames.
IPA: Intra-Prediction Angle.
IPM: Inter-Prediction Motion.
ITU-T: International Telecommunication Union - Telecommunication.
ISP: Internet Service Provider.
JSVM: Joint Scalable Video Model.
LTE: Long Term Evolution.
MSS: Maximum Segment Size.
MM: Multi-Dimensional Multi-Section.
MSE: Mean Squared Error.
MOS: Media Opinion Score.
MPD: Media Presentation Description.
MAPE: Mean Absolute Percentage Error.
MAOMDV: Modified Ad Hoc On-Demand Multipath Distance Vector.
MDSR: Modified Dynamic Source Routing.
MPEG: Moving Picture Experts Group.
MAODV: Multicast Ad Hoc On-Demand Distance Vector.
NZ: Non-Zero.
OSI: Open Systems Interconnection.
OSCAR: Optimized Stall-Cautious Adaptive BitRate.
PSNR: Peak Signal-to-Noise Ratio.
P-frames: Predicted Frames.
PU: Prediction Unit.
QoS: Quality of Service.
QP: Quantization Parameter.
QS: Quantization Steps.
QoE: Quality of Experience.
RANs: Radio Access Networks.
RD: Rate-Distortion.
Re: Rebuffering.
RFC: Request for Comments.
RMD: Rough Mode Decision.
RTT: Round-Trip Time.
SHVC: Scalable High Efficiency Video Coding Extension.
SVC: Scalable Video Coding.
SARA: Segment-Aware Rate Adaptation.
SACK: Selective Acknowledgment.
SS: Slow Start.
SDN: Software-Defined Networking.
SSP: Streaming Service Providers.
SVM: Support Vector Machine.
SVB: Specific Value of Buffer.
TI: Temporal Information.
TCP: Transmission Control Protocol.
TU: Transform Unit.

UDP: User Datagram Protocol.
VLCs: Variable-Length Codings.
VB: Video Buffer.
VL: Video Levels.
VMAF: Video Multimethod Assessment Fusion.
VQA: Video Quality Assessment.
VIF: Visual Information Fidelity

1. Introduction

This introductory chapter outlines the motivation behind the study, focusing on the challenges posed by the rapid expansion of internet access and the explosive growth of multimedia content, particularly video streaming services. It highlights the need for a robust, efficient, and quality-adaptive video transmission mechanism due to issues like packet loss, variable delays, and network oversaturation. The chapter introduces ASViS (Adaptive Scalable Video Streaming), a novel cross-layer solution designed to bridge the transport and application layers, offering a sophisticated algorithm for data transmission based on layer-discarding policy and deadline-sensitive criteria. ASViS aims to enhance video streaming quality and consistency over current methods by optimizing bandwidth usage and alleviating network congestion. The objectives section lays out the general aim to design a transport layer protocol that adjusts video stream rate to available bandwidth while maintaining high-quality video and outlines specific goals such as analyzing SVC layers impact on video quality, designing a deadline-sensitive curfew-based algorithm, and implementing machine learning techniques for automatic curfew adjustments.

1.1. Motivation

The current era of digital communication is marked by a rapid expansion of Internet access, a phenomenon propelled by the widespread adoption of advanced access technologies. This includes the deployment of fiber optic networks, evolution in radio access networks (RANs), proliferation of local wireless networks, and the advent of satellite broadband. However, these technologies, despite their advancements, introduce a spectrum of challenges.

In parallel, the world of multimedia content, especially video streaming services, has experienced explosive growth. Platforms offering pre-recorded and live video streams are now commonplace, yet this surge in multimedia content brings to the forefront numerous network-related challenges. These include increased packet loss rates, the stochastic nature of wireless channels, network oversaturation, unstable throughput, and variable delays. These issues severely impact the fidelity of the original content. Although the implementation of "media-friendly" and "TCP friendly rate control" has contributed to improved designs, the overwhelming growth of multimedia content necessitates a more robust, efficient flow-controlled, quality-adaptive video transmission mechanism [1].It is important to note that all streaming applications based on UDP should incorporate congestion avoidance mechanisms, as stated in RFC 8085, sec. 1, p. 4 [2].

Moreover, the intricacies of video content delivery, particularly those revolving around DASH and Adaptive Bitrate (ABR) algorithms, compound these challenges. Variability in frame bitrates, competition among diverse flows in the last mile of Internet Service Provider (ISP) networks, and the cascading effect of downloading high bitrate chunks that deplete the buffer are significant issues. The conservative nature of existing ABR algorithms, often designed to minimize rebuffering events, can result in suboptimal average bitrates and a consequent decline in video quality.

To further elaborate on the urgency and relevance of addressing the outlined challenges

in video streaming technologies, consider the global shift towards remote work, online education, and digital entertainment. This transformation has exponentially increased the demand for high-quality, reliable video streaming services. The COVID-19 pandemic has particularly underscored the importance of digital communication platforms, revealing significant gaps in current technologies when faced with unprecedented levels of internet traffic [3, 4]. For instance, the transition to online learning has highlighted the critical need for efficient, adaptive video streaming to ensure educational content is accessible to students in varying internet conditions [5, 6]. Similarly, the entertainment industry has seen a surge in online viewership, placing additional strain on existing network infrastructures. These scenarios illustrate the pressing need for innovative solutions that can adapt to fluctuating network conditions without compromising video quality or user experience. Addressing these challenges is not just about enhancing current technologies; it is about reimagining the future of digital communication to meet the evolving demands of a world increasingly reliant on virtual connections.

1.2. Hypothesis

This thesis posits that the introduction of ASViS will effectively address the critical challenges of increased packet loss rates, the stochastic nature of wireless channels, network oversaturation, unstable throughput, and variable delays identified in the realm of video streaming. ASViS, a novel cross-layer solution designed to operate between the transport and application layers, introduces a groundbreaking algorithm that manages data transmission based on a sophisticated layer-discarding policy and deadline-sensitive criteria. This marks a significant departure from traditional ABR algorithms that rely on TCP reliability mechanisms.

ASViS uniquely uses packet loss as a metric to calibrate flow rate, incorporating aspects of TCP methodology while avoiding retransmissions. Supported by a well-defined discarding policy, ASViS efficiently detects missing data at the application layer, such as absent frame fragments, utilizing UDP. This approach circumvents the need for sequential tracking and acknowledgments characteristic of TCP, thereby reducing associated delays. The research posits two primary hypotheses for validation:

- ASViS will significantly enhance video streaming quality and consistency over current DASH and ABR methods. By bridging transport and application layers with its innovative algorithm, ASViS is expected to reduce rebuffering incidents and elevate overall video quality in diverse network conditions.
- ASViS will optimize bandwidth usage and alleviate network congestion through its unique flow rate regulation approach. This UDP-based strategy will enable efficient data loss detection and facilitate informed retransmission decisions at the receiver, enhancing the end-user streaming experience.

These hypotheses frame the expected impact of ASViS in addressing the aforementioned challenges within the broader context of multimedia content delivery and network efficiency. Central to ASViS is its innovative method for flow rate regulation, akin to TCP congestion window, and the integration of a selective acknowledgment protocol, crucial for ensuring accurate RTT measurements and throughput estimation. ASViS, adhering to RFC 8085

guidelines, promises not only to mitigate network congestion but also to significantly enhance the quality of video streaming experiences, positioning it as a transformative solution in multimedia content delivery.

1.3. Objectives

1.3.1. General Objective

To enhance the streaming experience across fluctuating network conditions by optimizing the quality and efficiency of video transmissions. This objective aims to ensure the delivery of high-quality video content, improving reliability and user satisfaction by effectively managing variable bandwidth and minimizing video rebuffering and delays.

1.3.2. Specific Objectives

- Evaluate how different SVC layers influence the quality of video streaming to determine the optimal configuration that ensures the best user experience.
- Develop and refine algorithms that dynamically adjust video streaming to available bandwidth and network conditions, aiming to minimize buffering and enhance video quality.
- Integrate machine learning to automatically fine-tune streaming parameters in real time, aligning closely with changing network conditions to meet and exceed quality targets.
- Perform advanced simulations to validate and optimize the streaming protocol under various network scenarios, ensuring the protocol consistently delivers high-quality video.

2. Foundations and Innovations in Video Streaming Technologies

Over recent years, a marked transformation in the realm of digital communication has been observed, with multimedia video streaming emerging as the predominant internet data service. Some had predicted that by 2020, video traffic would account for 82% of global online traffic [7]. Contrary to these predictions, it is reported in [8] that in the first half of 2022, video streaming constituted 65.93% of total internet traffic, with Netflix and YouTube contributing 13.74% and 10.51%, respectively. Furthermore, a 23% increase in total traffic volume was observed in 2022 compared to 2021, attributed to the significant growth of various streaming services. Current data regarding video streaming consumption on mobile networks indicates that 71% of the traffic was video-related [9]. Projections suggest that this figure is expected to reach 80% by 2028. Such statistics underscore the undeniable growth trend. This evolution has been attributed not only to technological innovations but also to the integration of intelligent devices. Platforms, such as Zoom, Microsoft Teams, Netflix, YouTube, and Amazon Prime, have been recognized as significant players in this revolution. Moreover, unexpected external events, notably the Coronavirus disease pandemic, have acted as catalysts, propelling an unprecedented demand for online communication and entertainment platforms. This surge, documented as a 20-40% increase in video streaming [3, 4], occasionally strained available bandwidth capacities, necessitating temporary transitions from high definition to standard definition by some service providers [10].

DASH is an open standard for transmitting video that adapts to both the network and device. In this evolving landscape the DASH protocol, operating over the TCP, has been identified as the primary delivery model for video streaming services [11, 12, 13]. This open-source standard, celebrated for its flexibility and codec-agnostic features, has established itself as an indispensable tool for most streaming entities [14, 15]. DASH divides the video into multiple fixed-length chunks [16], each encoded at different qualities, frame rates, and resolutions. Nevertheless, challenges associated with DASH, including buffering interruptions, have signaled an imperative need for refined protocols that can mitigate such disruptions[12]. The most common problems are focused on coding, ABR, and congestion control, which are described below.

- Not all frames have the same bitrate due to differences in complexity in terms of motion or elements. Therefore, even with a constant bandwidth, it is necessary to consider bitrate oscillations [17, 18].

- In the last mile of the ISP, the video stream competes with many different flows, and since TCP transmissions are the majority of flows, this inevitably becomes a bottleneck. This leads to events such as packet loss and inflation of end-to-end RTT delays[19].

- Downloading certain high bitrate chunks can deplete the buffer, forcing the next chunks to have a low bitrate to maintain smoothness[20, 21]. This is known as the cascading effect.

- ABR algorithms have a conservative policy to avoid or minimize rebuffering events [21]. In [22], it shows the YouTube behavior to variations in bandwidth, particularly for wireless access, where the ABR algorithm does not increase the video quality until several seconds pass. These can lead to a lower average bitrate.

- Once the ABR algorithm chooses a certain bitrate to download, it cannot be canceled. Therefore, if during this interval, the network bandwidth fluctuates, and the buffer is empty, this can lead to rebufferings [23].

- Using ABR in conjunction with SVC makes the video clips independent, which forces downloading the clips with a higher bitrate to have better video quality.

- ABR algorithms have an expensive computation overhead, especially in the latest proposals [20, 21]. This implementation can be problematic, especially in the case of mobile devices that have limited computational resources [20]. That is relevant because, as of 2020, mobile devices are 43% of online video views [24]. That is one reason why ABR algorithms are generally implemented on the server side.

Through an ABR algorithm, the client fills the local disk with chunks sent from the server, stores the content before it starts to play, and helps tolerate network outages without interrupting the playback by using the stored content in the local disk [25]. For simplicity, the video content stored before the playout on the client side is called a video buffer (VB). Also, the VB definition helps to avoid confusion with the buffer term in the context of UDP and TCP transmission. The video player employs an ABR algorithm to select and switch the content quality based on instantaneous network conditions. The main objectives ABR algorithms are [25, 26, 27]:

1. Deliver the highest possible video quality.

2. Avoid rebuffering.

3. Minimize playout delay.

4. Minimize quality switches.

It may seem impractical to achieve all these objectives simultaneously; hence, ABR algorithms aim to strike a balance between these points and achieve the best possible Quality of Experience (QoE). From the perspective of the Streaming Service Providers (SSP), QoE is one of the most critical factors for measuring service satisfaction among users. This is because the content is now accessible on a wide range of devices and access points. Recent studies indicate that rebuffering, initial delay, and variations in video quality significantly impact QoE perception [25, 26, 28, 29]. The effects of these factors also depend on user subjectivity [27]. Therefore, the primary objective of the SSP is to maximize QoE. In an ideal situation, the SSP would deliver the highest possible video quality and framerate without any rebuffering. However, bandwidth is limited, and network conditions can fluctuate over time, with different technologies used to access the internet.

One of the traditional codecs that have been used in DASH in recent years is H.264 or Advanced Video Coding (AVC) [30, 31]. AVC provides various techniques and tools to create compression with loss. It exploits the similarity between adjacent pixels and the movement

of specific structures or objects in successive frames, creating a structure of dependent frames known as Group Of Pictures (GOP). With time, the need for video scalability led to SVC as an extension of AVC. SVC can encode videos using the same techniques as AVC, creating a Base Layer (BL) that can be independently decoded, and adds another layer called Enhancement Layer (EL), which depends on the BL [32, 33, 34]. EL provides scalability in terms of quality, framerate, and resolution. If an EL is lost, the BL can be independently decoded with basic video quality, which gives SVC better resilience to packet loss compared to AVC [33].

The structure of independent and dependent layers has been implemented in subsequent evolution and new encoding algorithms used by SSPs such as H.265 or High Efficiency Video Coding (HEVC) with its Scalable High Efficiency Video Coding extension (SHVC) [35, 36, 37], Video Predictor 9 (VP9) [38, 39] and AV1 [38, 40, 41]. However, these codecs face challenges such as high royalty payment for H.265 [42, 43], discontinuation of VP9 in favor of AV1 [40], and AV1 still maturing [38], even though it is slowly starting to gain popularity.

Advancements in the field of video streaming have been significantly influenced by the integration of artificial intelligence (AI). Video encoding has been enhanced through algorithms powered by AI, which dynamically adjust encoding parameters, resulting in optimized data compression while maintaining visual quality [44, 45, 46]. ABR algorithms have been refined through machine learning techniques, predicting network conditions to facilitate smoother streaming and reduced rebuffering [29, 47]. Furthermore, the assessment of video quality has been advanced by predictive models that effectively gauge user-perceived quality, enabling swift feedback for real-time adjustments [48, 49]. These AI-driven improvements have not only enhanced operational efficiency but have also enriched user experience by customizing streaming to meet individual demands and viewing contexts.

In parallel with AI advancements, the deployment of high-speed networks such as 5G and satellite networks like Starlink has broadened the horizons for video streaming [50, 51]. The high data transmission capacity and reduced latency associated with 5G have paved the way for high-definition streaming on mobile devices, whereas satellite networks have extended streaming reach to previously disconnected areas. Nonetheless, these advancements present unique challenges, including efficient broadband spectrum management and adaptation to increased network variability. Consequently, the development of solutions that optimize video streaming on these emerging platforms has been necessitated, ensuring that the technology can fulfill the promise of uninterrupted, global access to quality content.

In future research and development, ongoing optimization of ABR algorithms is seen as pivotal. As video encoding technologies advance and high-quality content demand increases, these algorithms must be refined to balance video quality, buffering minimization, and network variation adaptation more effectively. Furthermore, the integration of ABR with emerging networks like 5G is identified as a promising development area. The deployment of 5G promises significantly higher data transmission speeds and reduced latency, potentially revolutionizing video streaming on mobile devices and in previously inaccessible locations. However, challenges such as efficient broadband spectrum management and adaptation to increased network variability are presented by this integration, necessitating the development of solutions that leverage 5G capabilities fully. Such advancements are anticipated not only to enhance end-user streaming experience but also to open new avenues for interactive and real-time content, marking the beginning of a new era in digital video streaming.

2.1. Thesis Work Contribution

In this thesis, ASViS, an innovative algorithm for scalable and adaptable video transmission, has been introduced, marking a significant advancement in the field of Adaptive Bitrate (ABR) algorithms. The primary contribution of this research lies in the development and empirical validation of a theoretical model that optimizes video quality and minimizes transmission interruptions, even under fluctuating network conditions. Through a series of meticulous experiments, it has been demonstrated that ASViS surpasses conventional ABR methods, such as MPC, in terms of video quality, bandwidth efficiency, and transmission stability. Specifically, ASViS is shown to have a remarkable ability to adapt dynamically to network variations, prioritizing the transmission of critical video layers and proactively adjusting the sending rate in response to changing conditions. This approach not only improves user experience by reducing startup times and rebuffering but also suggests a more efficient utilization of network resources. The relevance of this thesis extends beyond technological advancements, proposing a model that can be integrated with current and future video encoding technologies, and providing a solid foundation for future research in video transmission optimization in next-generation networks like 5G.

2.2. Communications Protocols

2.2.1. Conceptual Network Layers Models

2.2.1.1. TCP/IP

TCP/IP refers to a set of data communication protocols, that was developed around 1973 as part of the Advanced Research Projects Agency Network project of the United States Department of Defense. In some cases, TCP/IP is called the Internet Protocol Suite. In general, it determines two things: first, how the computers are connected to the Internet, and second, how should be the information interchanged, describing how the data must be framed, encapsulated, packeted, encoded, transmitted, received, decoded, and, finally, used by the layers of the application [52]. Maybe the most relevant characteristics of TCP/IP are its simplicity and robustness. For that reason is considered a practical way to understand communications, which allows it to be implemented in many environments. The traditional scheme of TCP/IP is a protocol stack, in this, each layer is dependent on its top layer. This is described next [53, 54]:

- Application: This layer provides the interface, session establishment and coordination, and communication protocols between users and software applications.
- Transport: This layer ensures the transfer of data streams and flow control using two protocols: TCP guarantees data integrity by checking for corrupted data, while UDP does not check reliability.
- Internet: This layer is connectionless and responsible for exchanging messages between networks, defining data encapsulation, determining paths between networks, Internet addressing, packet delivery across multiple networks, and fragmentation and reassembly of packets.

- Data Link: This layer identifies network protocols, synchronizes frames with the Logical Link Control, performs connection, and defines permissions for data exchange through the Media Access Control. It adds header frames to packets.
- Physical: This is the lowest layer responsible for physical interfaces using cables or wireless signals, sending a series of 0s and 1s and controlling the bit rate.

2.2.1.2. Open Systems Interconnection (OSI)

The OSI model is more extended and refined than TCP/IP, it was proposed to be a common frame for new network kinds and protocols, allowing different networks can be interoperability. His major advantage is the agnostic model to the technologies or protocols used in the telecommunications process [52, 54]. The OSI model presents a 7 layers abstraction model, highly correlated with the TCP/IP model, Figure 2.1.

- Application: The closest to the users and the software, provides the high levels protocols to send and receive data
- Presentation: Defines how the data must be codified, encrypted, and compress data.
- Session: This layer is in charge of authentication, creation, maintenance, ensuring, and closing sessions between devices to ensure the correct data transference.
- Transport: Can divide the data into packets with a fixed length or reassembles the packets to extract the data. Is responsible, as well, for sending the data at a determined connection speed and finally for checking the integrity of the data.
- Network: It layer discovers the path across the physical network between two devices to route the packets, uses network addresses, and split the packets into packets with a fixed length. It is very similar to the TCP/IP model.
- Datalink: It layer is equal to the TCP/IP model.
- Physical: It layer is equal to the TCP/IP model.

2.2.2. TCP Congestion Control

At the beginning of the Internet, the foreseeable growth of telecommunications and the need for methods that will allow reliable communication gave way to the TCP part of the TCP/IP model. TCP provides a way to be aware of network capacity variations and was developed with intrinsical congestion control to prevent it.

The first formal version of TCP was TCP Tahoe in 1988, which included Slow Start (SS), Congestion Avoidance (CA), and Fast Retransmit (FRT). TCP Reno was developed a few years later and included all mechanisms from TCP Tahoe as well as a new congestion control technique called Fast Recovery (FRC). TCP Vegas is another version that includes a new method to detect the congestion based on the packet delay (instead of the packet loss). Posteriorly TCP New Reno modified the FRC mechanism [55, 56]. The evolution of TCP can be seen in Figure 2.2.

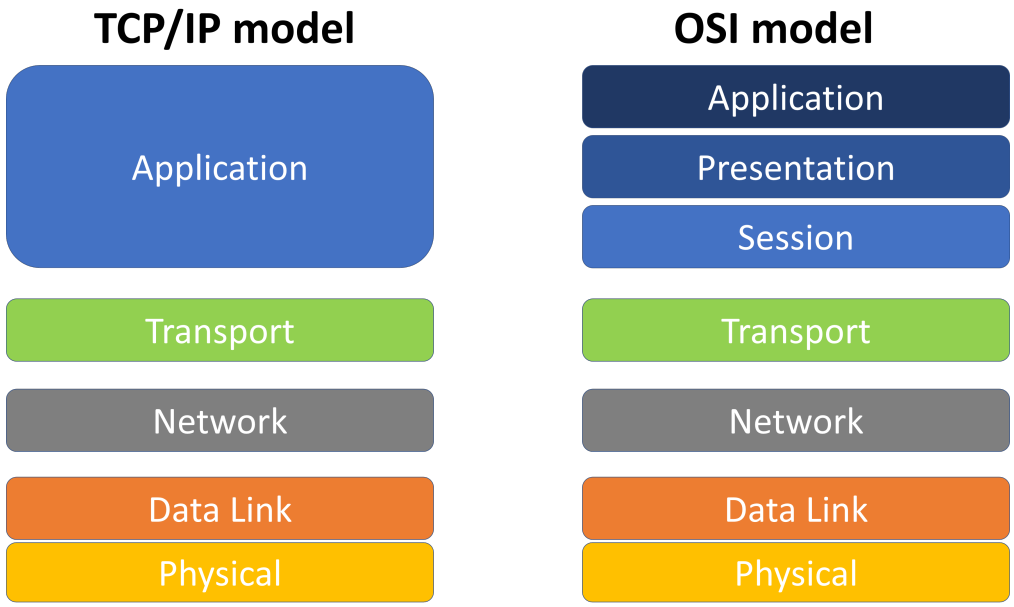


Figure 2.1: TCP/IP OSI models comparison.

TCP uses cumulative Acknowledgments (ACK) to identify if the packets were successfully received on the receiver side or not and to know that it is expecting the next one, if the ACK is undetected, the packets are resent. The Congestion Window ($cwnd$) is the number of packets that TCP can send before receiving an ACK, which helps to avoid overcharge data between the sender and the receiver. Duplicated ACK is sent when the receiver receives an out-of-order packet. The first step is SS, the $cwnd$ grows exponentially after receiving an ACK, that growth can be stopped in 2 situations [55, 57]:

- If a SS threshold ($ssthresh$) is reached the algorithm changes to the CA phase, in that the growth is linear after each RTT.
- When the sender receives 3 Duplicated ACK it assumes that the packets are lost, so changes to the FRC with a $ssthresh = cwnd$ and $cwnd = ssthresh + 3$ and adopts a linear growth after each RTT, at this stage, it only resends the packages that were lost. After that, the algorithm changes to CA, maintains the $ssthresh$, and $cwnd = ssthresh$.

When TCP RENO detects a timeout, it change to SS using the previous $cwnd$ to define a new $ssthresh = cwnd/2$. The Additive Increase/Multiplicative Decrease (AIMD) behavior is the core in terms of congestion control because it provides the ability to react quickly against congestion events.

2.3. Video Codification

Uncompressed video is commonly referred to as RAW video. It is essentially a sequence of images that have not been compressed or processed in any way. In order to achieve higher compression ratios and reduce the bit rate of RAW videos, video coding techniques are employed. This saves bandwidth and space required for data storage. The video coding

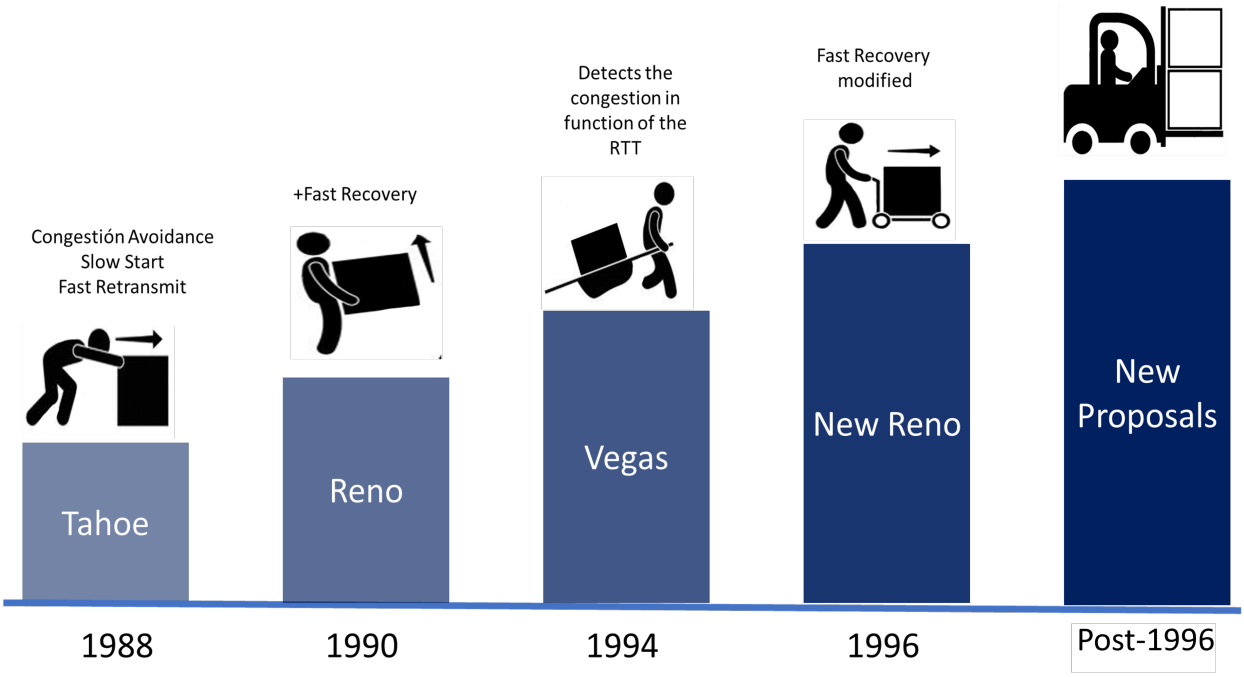


Figure 2.2: Evolution of TCP protocols.

process involves three main steps: redundancy elimination, quantization, and coding of the coefficients. The two most widely used standards in video coding are the Moving Picture Experts Group (MPEG) 4 part 10, also known as H.264, and its successor, HEVC or H.265.

Furthermore, video coding plays a crucial role in the streaming and transmission of videos over the internet. The use of video coding algorithms enables the efficient transmission of high-quality videos without consuming excessive bandwidth. However, the constant evolution of video coding standards is necessary to keep up with the ever-increasing demand for better video quality and efficiency.

2.3.1. H.264

The H.264 video coding standard, also known as AVC, was designed by the Joint Video Team (JVT) in 2003 [31] and is one of the most widely used and sophisticated video compression solutions. It consists of four main steps: motion estimation, transform coding, quantization, and entropy coding [58].

The motion estimation process eliminates redundancy by utilizing spatial and temporal correlation between neighboring pixels and successive video frames. Each video frame is divided into macroblocks of size 16×16, which can be further divided into smaller blocks (16x8, 8x16, 8x8, 8x4, 4x8, and 4x4) for motion estimation of moving objects [58, 31]. Larger macroblocks are suitable for homogeneous movement, while smaller blocks are useful for fast movement or detailed objects. Using this information, a new macroblock is predicted by spatial or temporal prediction.

The transform coding process involves converting a block of NxN pixels into a block of NxN coefficients using a Discrete Cosine Transform (DCT). This process groups the energy of the image into a few significant values, with high-frequency coefficients tending to zero and entire coefficients obtained. Zigzag exploration is used to extract relevant data, with the

highest value being found at low frequencies near the upper-left edge of the block.

Quantization is a lossy compression method used to encode the DCT coefficients by removing non-significant data. H.264 has 52 different quantization steps (QS), each of which increases the quantization level by 12.5%, providing fine control between quality and size. The quantizers are selected by the Quantization Parameter (QP). Equation 2.1 shows a simplified version of quantization, where Z is the quantized coefficient, Q_{step} is the QS, and Y are the coefficients to transform. Different step levels are used according to the desired quality, and the result is rearranged into arrays by zigzag, ensuring the highest energy coefficients are in the first place.

$$Z = round\left(\frac{Y}{Q_{step}}\right) \tag{2.1}$$

Entropy coding creates an adequate bit flow for transmission and can use two alternatives: Context-Adaptive Variable Length Coding (CAVLC) and Context Adaptive Binary Arithmetic Coding (CABAC).

- CAVLC is designed to exploit the characteristics of non-zero (NZ) DCT coefficients. After the quantization process, the blocks mostly contain zeros, and the zero strings can be compactly represented by run-level coding. Conversely, the frequency of NZ coefficients is high, and they are typically ± 1. CAVLC uses seven fixed variable-length codings (VLCs) to encode these NZ coefficients. Each table has a high range of values, and the same NZ string can be coded differently with different VLC tables. CAVLC uses two modes for coding: the regular mode and the escape mode. The regular mode is used when the NZ string values are within the range of the table; otherwise, the escape mode is used. The transition threshold for the next NZ string is adaptive and is based on the current NZ string values.

- Although CABAC is more efficient than CAVLC, it is complex and expensive in processing. The encoding process has three steps: binarization, context modeling, and binary arithmetic coding. In the binarization step, non-integer coefficients are mapped to a binary sequence. This step uses a lookup table to obtain a fixed symbol rate for each symbol of an alphabet. In the context modeling step, the NZ coefficients previously coded are used to estimate the conditional probabilities of the next NZ coefficients to be coded. Finally, in the binary arithmetic coding step, the symbol is encoded according to the conditional probabilities model.

2.3.1.1. Scalable Video Coding

SVC is an extension of the H.264/AVC standard, developed by the International Telecommunication Union - Telecommunication (ITU-T) H-series recommendations for the coding of moving video. It allows for the generation of multilayer streams from a single layer coding, which in turn increases the complexity of the encoding/decoding process. SVC has the unique characteristic of partitioning the compressed video frames into distinct incremental layers, which can be temporal, spatial, or quality-based [59]. Each substream obtained by dropping packets from the original bitstream results in a lower frame rate, lower resolution, or lower quality video signal. These substreams are categorized into BL and EL, with the BL containing independent layers and the EL containing dependent layers.

The layers are combined to form a single video stream, which should not be confused with a solution consisting of multiple distinct-quality copies semaphored by a throughput-sensing system. The SVC coding mechanism partitions the spatial frequency information into multiple packets, where the lowest frequencies form the base layer and subsequent frequencies form the upper layers. The upper layers are useless without the lower layers, as the higher frequencies do not contribute significant information without the lower frequencies. The highest layer, which contains the highest frequency content, will appear as noise without the remaining data. For this reason, the decoder discards higher-level layers if any intermediate layer is missing.

In the context of video streaming, scalability refers to packeted layered coding that supports the discarding of higher layers in a manner that is still decodable. This increases the granularity of the packed packets, thereby losing less information if packets are lost. In contrast, for the single-layer case, if a whole frame is sent partitioned in standard-sized packets and a single packet is lost, the affected frame would be incomplete and therefore, undecodable. The temporal scalability of SVC supports the decimation of bi-directional frames with dependencies, forming another valid bitstream that represents the source content with a lower frame rate. The quality scalability enables the reconstruction of substreams by removing higher spatial frequency content [30].

2.3.2. H.265

H.265, also known as MPEG-H Part 2, was developed by the Video Coding Experts Group and the MPEG to meet the growing demand for video consumption, likely due to the increasing availability of high-speed internet and streaming services like Amazon Prime, Netflix, and YouTube. HEVC supports Full Ultra High Definition (UHD) and offers over 50% improvement compared to its predecessor [60, 61, 62].

In image partitioning, HEVC uses the Coding Tree Unit (CTU), which is a logical unit composed of three blocks: luminance and two chromas (Cn and Cr), as well as an associated element, as shown in Figure 2.3 [60]. While CTUs are similar to the macroblocks in H.264, they are larger and more flexible. Each CTU contains between 16x16 and 64x64 pixels [63].

CTUs can be divided into smaller units down to 8x8, and these divisions are called Coding Units (CU). The CU consists of two units: the Transform Unit (TU) and the Prediction Unit (PU). The TU contains information on the Discrete Fourier Transform (DFT) matrix and quantized blocks, while the PU stores prediction data for either Intra-Prediction Angle (IPA) or Inter-Prediction Motion (IPM). The CU and PU can range from a maximum size of 32x32 to a minimum of 4x4 pixels.

For the motion estimation process, HEVC can use the inter prediction mode and introduce the merge mode, which creates a motion vectors region by copying temporal and spatial axis information. Later, a motion vector machine selects the best motion vector from a list of candidate vectors. The sizes of the motion vector prediction are very flexible and can be asymmetric or symmetric. The motion vector prediction uses two main references, L0 and L1, each with 16 references.

The motion estimation process can also use the intra-frame prediction, which consists of three steps: Rough Mode Decision (RMD), Most Probable Mode (MPM), and Rate-Distortion (RD) optimization. RMD selects an initial best candidate list from 35 predictors, MPM adds the coding modes of above and to the left of the current PU to the list, and RD optimization chooses the minimum RD cost in the candidate list as the best coding mode.

These trade-offs between distortion (objective video quality) and bit rate require an extensive amount of processing.

For the transformation process to the frequency domain, HEVC uses DCT to eliminate spatial redundancy. HEVC can use many transforms, including 8x8, 16x16, 32x32, and 4x4. For intra-frame prediction, the 4x4 mode is based on Discrete Sine Transform (DST). In general, CABAC remains essentially unchanged but introduces enhancements in coding efficiency and computational complexity compared to H.264. CABAC improves compression efficiency by 9% to 14% [64]. However, the support for CAVLC is abandoned.

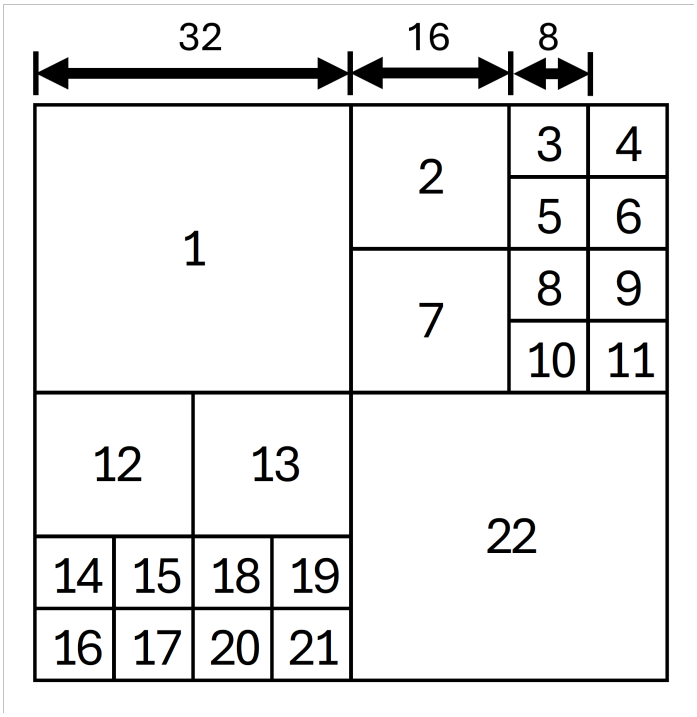


Figure 2.3: Partitioning Structure of the Coding Tree Unit (CTU) in HEVC.

2.3.2.1. SHVC

SHVC is a video coding standard that has been developed as an extension to the HEVC standard to provide enhanced video quality, higher bit depths, wider color gamuts, and higher frame rates. The primary advantage of SHVC over HEVC is its scalability, achieved by encoding one video in multiple layers, where each layer provides a specific enhancement[35]. This scalability can be achieved in three ways: temporal, spatial, and quality. Temporal scalability allows control over the frame rate, spatial scalability controls the spatial resolution, and quality scalability manipulates the fidelity of the video [36, 37].

The ability to adapt to network conditions is another major advantage of utilizing SVC or SHVC. Network conditions can change rapidly, and video quality can suffer if the codec is not designed to handle such changes. With SVC and SHVC, the video can adapt to network conditions, resulting in a better viewing experience for users. This ability to adapt also makes SHVC an ideal choice for streaming high-quality videos over the internet or other networks[35].

One of the principal novelties of SHVC is transcoding, which is the process of reencoding a video with different parameters. This technique is useful when the video needs to be converted to a different format or compressed to a lower bit rate to save storage space or reduce network bandwidth. Another novelty of SHVC is simulcast, which is the process of encoding multiple videos, each with a different resolution, frame rate, bit depth, etc., and transmitting them simultaneously [36, 37]. This allows viewers to choose the version of the video that best suits the capabilities of their devices, network speed, or personal preference.

In summary, SHVC is an advanced video coding standard that offers scalability, adaptability, and advanced features like transcoding and simulcast. These features make SHVC an ideal choice for applications that require high-quality video and the ability to adapt to changing network conditions. The main differences between SVC and SHVC are described in table 2.1.

Table 2.1: Comparison of the scalable coding features between H.264 and H.265 (adapted from [35]).

Scalability features	**Standard**	
	SVC	**SHVC**
Temporal	x	x
Spatial	x	x
Quality	x	x
Hybrid codec		x
Bit depth		x
Color gamut		x
Simulcast		x

2.3.3. Group of Pictures

GOP is a set of consecutive frames in a coded video stream, and at the same time, it is the structure in which the predicted frames are arranged. The compression process can lead to two types of prediction based on the similarity between adjacent pixels (spatial) and the motion of objects (temporal) [65].

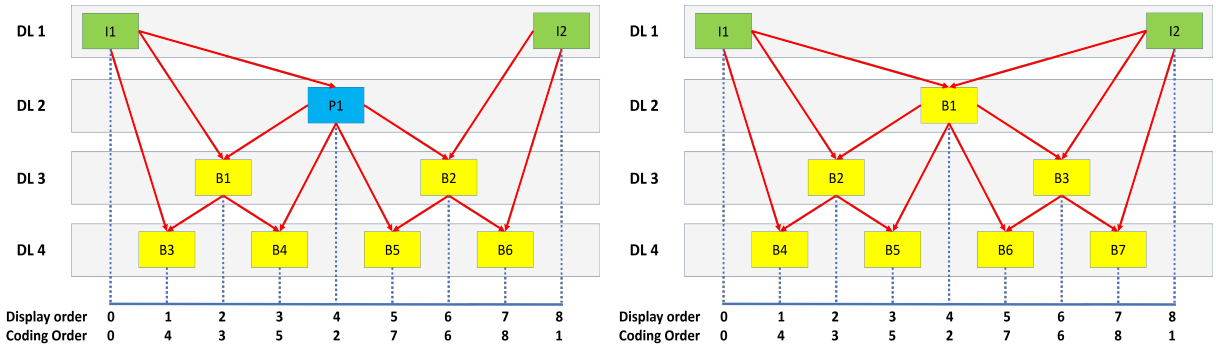
- Intra prediction involves creating a predicted macroblock with the adjacent macroblocks and coding the residual information between the predicted and the current macroblock.
- Inter prediction involves finding similarities between previously encoded macroblocks (k) and the current one (k+1), creating a motion vector that indicates the position of macroblock k on k+1.

These two types of prediction, intra and inter, can be arranged into three different types of frames, which are related and may be necessary to decode the others [12, 66, 67].

- Intra-frames or I-frames: These frames contain all the information of the original image, can be decoded independently, and only contain intra-predicted macroblocks.
- Bi-predicted frames or B-frames: These frames are based on the previous and subsequent frames and only contain inter-predicted macroblocks.

- Predicted frames or P-frames: These frames are based on the previous frames and contain both intra-predicted and inter-predicted macroblocks.

The GOP must contain at least one I-frame, which has the least amount of compression. A GOP can contain P-frames and B-frames. P-frames require one I-frame or P-frame to decode the data. B-frames require two frames (preceding and succeeding in display order) of any type to decode the data. The scenario with I-P-B frames is illustrated in Figure 2.4a, while the scenario with I-B frames is illustrated in Figure 2.4b .



(a) (b)

Figure 2.4: Hierarchical structure of GOP.

The GOP size is a manipulable parameter that indirectly defines the size of the frames because there exists a dependency between frames to decode certain data. In this case, we refer to this as the Dependency Level (DL). In Figure 2.4, frames at the highest DL have the highest frame size, and descending in the DL also decreases the frame size. However, the frame size depends on a series of characteristics of both the codec and pictures, so this DL is not a linear relation.

In the SVC context, we can differentiate between Base Layer Interpredicted (BL-I), Enhancement Layer Interpredicted (EL-I), Base Layer Bi-predicted (BL-B), Enhancement Layer Bi-predicted (EL-B), Base Layer Predicted (BL-P), and Enhancement Layer Predicted (EL-P). Figure 2.5 depicts the GOP structure when SVC coding is considered. It considers a GOP size of 8 frames (IBBBBBBB) and two quality layers (base and enhancement). The B-frame decoding dependencies are also illustrated, and P-frames are not considered in this scenario. These dependencies are important as they explain how a missing frame at the decoder will impact the video quality. For instance, if frame 3 (encoding order) is missing, it will impact the decoding of frames 4 and 5, resulting in artifacts. This behavior is called *error propagation*. If frame 5 is missing, there is no error propagation, as this frame does not serve as a reference to any other frame.

2.3.4. Video Quality Measure

Defining a video quality measure is a controversial topic, as concepts such as *good*, *fair* or *poor* quality depend on a series of subjective factors. Attempting to translate the subjective measure into an objective one is complicated. Furthermore, the images that compose a video are degraded during several stages such as capture, quantization, compression, and transmission, among others[68, 69].

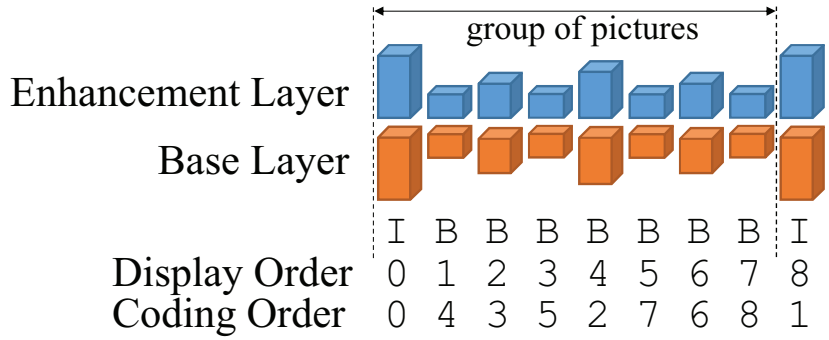


Figure 2.5: Display and coding order to GOP of SVC codification.

Despite the aforementioned difficulties, the main idea behind defining a video quality measure is to replicate how the human eye can transform light into biological signals. Photoreceptors are an essential compound of the retina that react to light, converting it into biological signals, and consequently, help us to obtain a visual representation of the world around us. In general, photoreceptors are classified into rod cells, associated with luminance recognition, and cone cells, associated with color recognition. The human eye has a significant difference in the total amount of photoreceptor kinds: rod cells are around 120 million, representing approximately 95% of the total, and cone cells are around 6-7 million, accounting for approximately 5%[70, 71].

Due to the reasons mentioned above, the human eye is more sensitive to brightness than color, and frequently, video quality assessment tools consider only the brightness components, i.e., the luminance or luma channel. The existing tools find an automated way to emulate human perception. Despite this, the subjectivity involved in quality assessment makes it challenging to develop an algorithm that provides an objective measure of video quality.

2.3.4.1. Peak Signal-to-Noise Ratio

PSNR is a widely used metric for measuring video quality, especially in DASH, and is one of the most common approaches for measuring QoE [18, 72]. One example is shown in Figure 2.6 [73], where Netflix encoded 100 different 1080p videos using h.264 with the same QP rate control, and plotted the resulting bitrate [$kbps$] on the x-axis and the Y-PSNR [dB] on the y-axis.

PSNR is the ratio between the maximum energy of a signal and the noise, expressed in [dB]. Although PSNR can calculate the components of luminance and chrominance, the most widely used parameter is luminance (Y), as the human eye is more sensitive to brightness[69, 74]. The first step in obtaining PSNR is to calculate the Mean Squared Error (MSE), given by [75]:

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2 \tag{2.2}$$

where $I(i,j)$ and $K(i,j)$ are the pixel values of the original and reconstructed image, respectively, and m and n are the sizes of the video. PSNR can then be calculated using Equation 2.3, where MAX_I is the maximum pixel value of the frame:

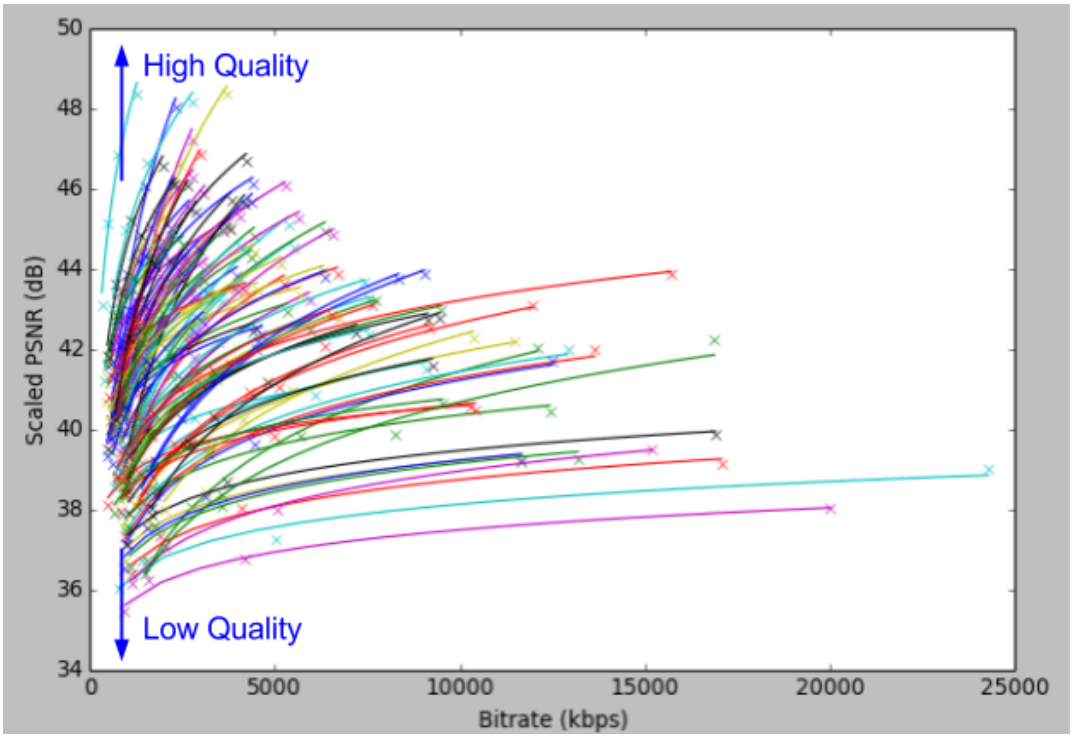


Figure 2.6: Netflix experiment analyzing PSNR vs bitrate for 100 videos which a resolution of 1080p (adapted from [73]).

$$PSNR = 10log_{10}\left(\frac{MAX_I^2}{MSE}\right) \tag{2.3}$$

Although PSNR is a widely used metric, there are problems associated with transferring a value in $[dB]$ to a subjective assessment. A more interesting approach is the Media Opinion Score (MOS), which is a subjective estimate of video quality and is represented by an average quality judgment of a group of testing subjects. The most common experiment to measure MOS involves having subjects watch a video and rate the perceived quality on a scale of 1 to 5. Although this experiment is expensive and time-consuming, a relationship was established between Y-PSNR and MOS, as shown in Table 2.2 [76], providing a good estimation of the perception of video quality across five categories: excellent, good, fair, poor, and bad.

Table 2.2: Mapping Y-PSNR to MOS (adapted from [76]).

Y-PSNR (dB)	**MOS**
>37	Excellent
31-37	Good
25-31	Fair
20-25	Poor
<20	Bad

While PSNR has been a foundational metric in video quality assessment, its capability to fully represent the subjective perception of end users has been a topic of scrutiny. This is due to its focus on mathematical differences at the pixel level, which does not always align

with human perception. Especially in adaptive streaming systems, there is a pressing need for metrics that can more accurately reflect human experiences.

2.3.4.2. Video Multimethod Assessment Fusion

In response to these limitations, Netflix introduced VMAF, a video quality metric crafted to mirror human perception more accurately [69, 77]. By amalgamating various video quality evaluation methods using a support vector machine (SVM), VMAF produces a perceptual quality score [29, 68, 78]. This integrated approach ensures the holistic representation of individual measures, bolstering its correlation with subjective assessments. Validations across a multitude of video resolutions and datasets indicate VMAF scores, ranging between 0 and 100, align closely with subjective evaluations [78, 79].

The first step in VMAF is to extract two types of video quality measures, spatial and temporal, as feature maps [78]. The spatial information is obtained by the mean detail loss metric (DLM) and visual information fidelity (VIF).The temporal measure is obtained using temporal information (TI). The DLM feature calculates the detail losses weighted over four different scales. Then, VIF obtains the losses and visual information fidelity and computes them into four different scales. The TI feature compares the luminance compound difference between a pair of frames to capture temporal effects due to movement and is quantified by six different features [78]. The average value of each feature is then obtained and used to feed a pre-trained SVM model, which obtains a predicted per-frame quality score from 0 for low quality to 100 for the highest quality [79].

The VMAF model was trained by SVM regression with a video database VMAF+, it is a large video quality assessment (VQA) dataset composed of 522 videos with different levels of scaling and compression [80]. In the supervised learning step, the VMAF model obtains suitable weights for the elementary metrics and learns from ground truth scores [79]. VMAF, equal to other state of art video quality measures, only analyzes the luma compounds of the video channel [80]. The outline of the VMAF algorithm is shown in Figure 2.7.

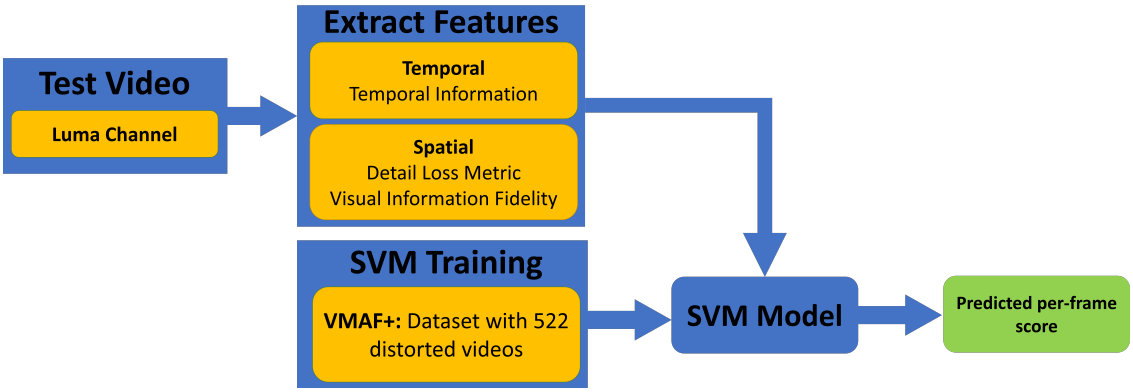


Figure 2.7: Outline of the VMAF algorithm.

2.4. Machine Learning

Machine Learning is a branch of artificial intelligence that studies techniques to perform tasks using learned experience as opposed to explicitly programming an inflexible input-

output function. There are different types of machine learning: supervised, unsupervised, and reinforced learning.

Supervised learning is mainly concerned with data regression and classification based on a labeled database. The system is fed the labeled data and the output is a correlation (numerical or class) of the newly acquired information and the previously collected data. Unsupervised learning is mainly concerned with association and clusterization. In this case, there is no labeled data, and the techniques are oriented to find patterns or similar traits. Reinforced learning is, in short, a system that performs a task by trial and error. It is given an objective and taught to distinguish a desirable outcome and by exhaustive attempts, it searches for a specified goal.

The integration of Machine Learning in the DASH standard, has been revolutionizing the delivery and quality of these services. For instance, segment prefetching has been addressed in a notable study, where ML predictions are utilized to adjust video segment bitrates to changing network conditions, achieving nearly 90% prediction accuracy and significantly enhancing user experience and bandwidth utilization [81]. Additionally, an innovative approach has been employed, utilizing Multi-access Edge Computing alongside ML classification models to select media segments for prefetching, thereby improving QoS and QoE in 5G network environments [82]. Moreover, a method based on Deep and Reinforcement Learning optimizes QoE by maintaining consistent video quality, demonstrating substantial improvements in wireless network environments [83]. Finally, with the advent of 5G communication technologies, another study in [84] has used ML models to estimate key QoE indicators from network-level QoS metrics, achieving a QoE prediction accuracy above 91%. These advancements underscore the capacity of ML to enhance both technical efficiency and user experience in the realm of video streaming . Because of the nature of the task, the focus of this work is centered on Supervised Learning. The technique that is most appealing to us is Supervised Regression. Explained ahead on 3.2.2.

2.5. Related Work

2.5.1. Previous work

This work builds upon [32], where the authors proposed a rate control algorithm that adjusts the transmission rate by discarding parts of a scalable video bitstream over TCP. The cross-layer interaction between SVC and TCP, along with a discarding policy based on different thresholds, enables the system to react quickly to network changes, reduce unnecessary traffic, and prevent unnecessary frame discarding. Many components of the proposed protocol are inherited from this work. However, some differences include reducing the variance of the congestion window to improve flow control smoothness, further protecting I-frames, introducing a dynamic curfew, theoretical modeling, a detailed analysis of quantization level effects, layer size analysis to minimize the number of packets used, and delay likelihood analysis based on curfew parameters, among others. The use of SVC and the discarding policy (explained in 3.2) are inherited from [32]. This work laid the foundation for the current work.

2.5.2. DASH

Video streaming involves distributing video and audio content over a network of computers. Typically, the video content is stored in a buffer before playback begins, and the buffer needs to be large enough to hold one GOP due to the relationship between decoding order and display order. Each GOP can be independently decoded without the need for previous or later GOPs.

In recent years, DASH has become the core of the SSP [20]. It was developed to provide robustness, flexibility, and avoid market fragmentation [72, 85]. DASH provides only general specifications for available multimedia content and how it should be packaged, along with a series of best practices. The logical implementations are typically provided by third parties. The DASH architecture is illustrated in Figure 2.8 [86]).

Each video is divided into multiple packets, each containing between 1 to 10 seconds of content. Each packet is encoded at multiple bitrates or qualities, referred to as Video Levels (VL) or chunks. A Media Presentation Description (MPD) is an Extensible Markup Language (XML) document that describes each chunk, and both the MPD and chunks are hosted on the DASH server. When a connection is established, the client receives the MPD, and information about the codec, quality, bitrates, chunk size, and other details stored on the server. On the client-side, DASH implements bitrate adaptation logic through an adaptation engine. The client continuously measures criteria such as buffer playout level, battery levels, Central Processing Unit (CPU) occupation, available bandwidth, etc., to select the most suitable chunks to download. The client then sends delivery requests and gets chunks via HTTP messages [18, 21, 86].

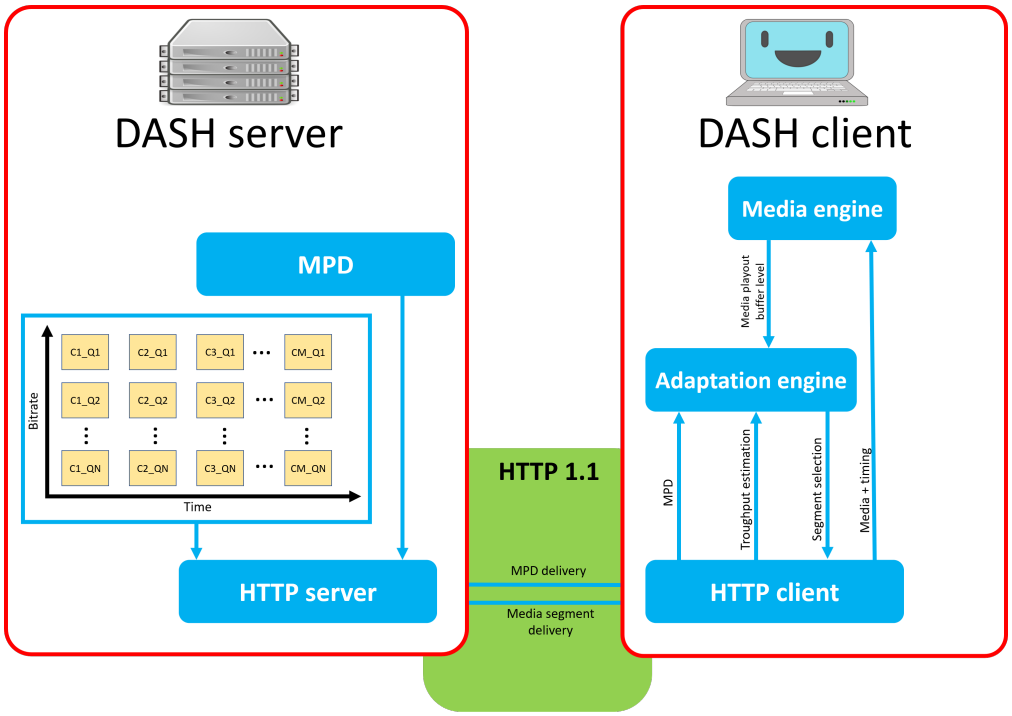


Figure 2.8: DASH architecture (adapted from [86]).

DASH has some intrinsic complications. The standard does not specify how to adapt the VL efficiently to the available bandwidth in real-time, and most streaming technologies

work on the best-effort model [87]. DASH attempts to send chunks at the highest resolution possible while also limited by network bandwidth. When the user perceives switches of VL or/and rebuffering, their QoE decreases.

Furthermore, in recent years, Radio Access Network (RAN) technologies such as Long Term Evolution (LTE), LTE Advanced, and most recently 5G have become more common. However, RAN has several intrinsic characteristics such as latency, packet drops and losses, handover, jitter, multipath, etc., that make it problematic to work with DASH. The above-mentioned reasons highlight the complications and limitations of DASH. However, the core of DASH is its ABR algorithms, which constantly monitor a series of network characteristics to deliver smooth playback.

2.5.2.1. ABR Algorithms

The primary objective of dynamic algorithms for delivering multimedia content is to minimize rebuffering events, maximize video quality by sending chunks with the highest possible quality, and reduce changes in the viewing experience [28]. In summary, an efficient adaptation of video quality to the available bandwidth in real-time.

In general, ABR algorithms need to ensure a certain level of buffer occupancy to avoid buffer overflow or an empty buffer. This is one of the major challenges in accurately predicting when to send chunks, as these chunks are downloaded before the video buffer content is displayed. Another pending challenge is that the network changes dynamically, and making a throughput estimation based on previous conditions is complicated.

Traditional measures such as packet loss, jitter, delay, or available throughput do not suffice to accurately reflect the experience of the viewer [28]. It is necessary to consider requirements such as avoiding cascading effects, maintaining smoothness in video quality during playout, and avoiding rebuffering events, but as mentioned in Section Foundations and Innovations in Video Streaming Technologies, achieving these goals at the same time is intrinsically contradictory.

One client can require the highest quality/bitrate chunks even if it does not have enough bandwidth, leading to rebuffering events. In a similar situation, but in a network with fluctuations (such as a typical wireless connection), choosing the highest quality can lead to quality switching and cascading effects, even if there is not enough bandwidth. Additionally, ABR algorithms are intrinsically conservative, avoiding switching video quality when conditions are favorable [21, 22]. In some cases, the available throughput is almost enough for a certain bitrate but well above the next available bitrate, so the algorithm needs to consider a risk trade-off between smoothness or rebuffering [21].

The literature presents many proposals to enhance the performance of streaming solutions. In general, the approach of the ABR algorithm can be classified as throughput-based, buffer-based, or hybrid-based [27, 88]. We present some of the proposals to enhance ABR algorithms, which can be seen in Figure 2.9.

2.5.2.1.1. Throughput-based algorithms

This approach monitors the instantaneous throughput while receiving chunks and smooths the throughput to compensate for short-time network fluctuations, thus avoiding estimation errors. It requests the next chunks with a bitrate equal or lesser to the measured throughput [23, 89, 90]. However, due to the time-variant nature of the network, the available capacity

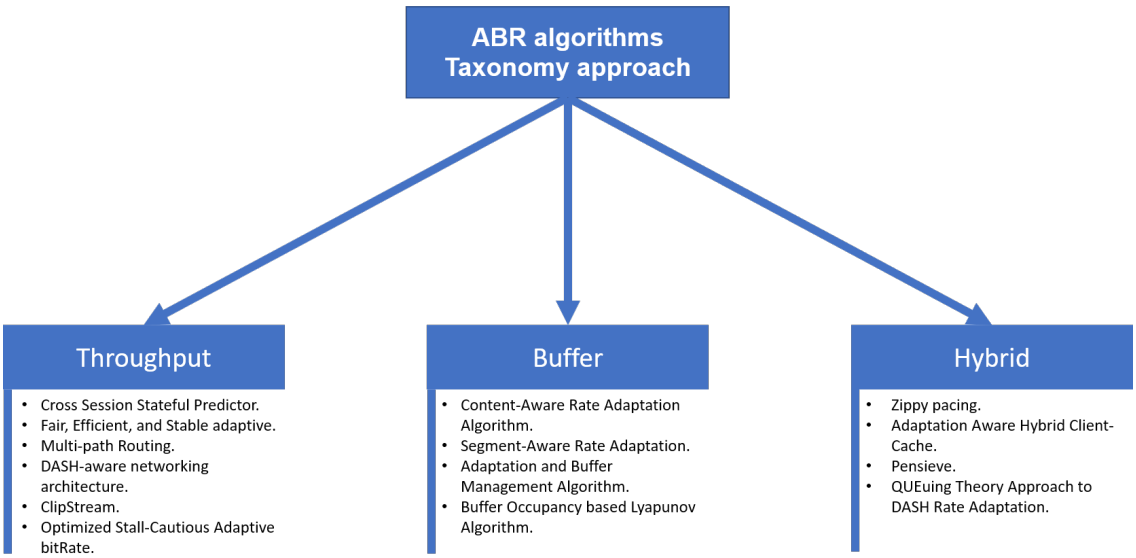


Figure 2.9: Taxonomy of different ABR algorithms approach.

can be under or overestimated, especially when the throughput-based approach is used with TCP because of its AIMD characteristic [89]. Moreover, network capacity estimation is often done above the HTTP layer, leading to imprecise information [89].

In [91], researchers proposed the Cross Session Stateful Predictor (CS2P) by analyzing a dataset of 20 million sessions and discovering that sessions with similar key features exhibit similar available bitrate and variability behavior. They created and trained the CS2P with this information, which yielded better results than the ABR algorithm used by the iQiyi SSP, in terms of average bitrate and QoE. However, the proposed algorithm requires the implementation of MPC [92], specifically the FastMPC implementation, whose complexity can make it difficult to implement on the client side and can cause a computational overload on the server-side. Fair, Efficient, and Stable adapTIVE (FESTIVE) algorithm was designed with a throughput-based approach and is based on the principle that multiple DASH players using the same link can generate a bottleneck that leads to fairness, efficiency, and stability issues [27]. FESTIVE consists of three components: bandwidth estimation, bitrate selection, and chunk scheduling. Additionally, it presents a general framework for robust video adaptation based on a randomized chunk scheduling, stateful bitrate selection, delayed update approach, and the harmonic mean of the throughput of the last 20 seconds.

Multipath routing is a solution for congested networks, where video content is spread from a source node to a destination node over multiple paths through the network [93]. To achieve this, routing protocols such as modified dynamic source routing (MDSR), modified ad hoc on-demand multipath distance vector (MAOMDV), multicast ad hoc on-demand distance vector (MAODV), etc. are used. The main concepts of multipath routing protocols are computing multiple paths between an origin and destination, securing data integrity with a multipath forwarding algorithm, and an end-host protocol to ensure the use of multiple paths. However, the biggest challenge in implementing multipath routing is configuring networks with quality assurance, which can be problematic on a large scale.

Reference [18, 94] proposes an approach to solving the problems caused by the mismatch between TCP and the adaptive bursty nature of DASH traffic using a networking architecture based on software-defined networking (SDN) with specific support for DASH streaming. Net-

work application controllers are used to assist DASH clients in bitrate selection and network management. A dynamic queue-based mechanism for Quality of Service (QoS) provisioning is used, and network controllers provide a complete overview of network conditions and can signal target bitrates to DASH clients while maintaining dynamic traffic control in the network, ensuring stable and high-quality video delivery.

Other solutions, such as ClipStream [95], rely on a throughput-based approach and stream over UDP, offering similar properties to TCP at the application layer. The idea behind ClipStream is that not all frames have the same importance; I-frames are more relevant than B-frames. Therefore, ClipStream uses reliable transport for I-frames and unreliable transport for B-frames. In the presence of losses in the unreliable stream, ClipStream employs Forward Error Correction (FEC) to recover lost packets. While ClipStream shows good performance, its implementation relies on UDP, which can be problematic to pass through firewalls or Network Address Translation.

Reference [96] presents the Optimized Stall-Cautious Adaptive bitRate (OSCAR) algorithm, which adapts to throughput variations and video quality in a probabilistic framework. Throughput variations are modeled as a random variable and are used to estimate a packet stall probability. OSCAR focuses on using a sliding look-ahead window for the future when selecting the quality chunk to adapt proactively to the available bitrate. One of the drawbacks of the OSCAR algorithm is that it is a multivariable problem, making it challenging to implement on a large scale.

2.5.2.1.2. Buffer-based algorithms

This approach aims to manage throughput variability and ensure frame deadlines. Clients download certain chunks and store them in a buffer before playout. To manage this, a buffer threshold is used, and when the playback buffer level is above it, the client can request a better video bitrate from the server. Conversely, when the playback buffer level is low, clients need to request a lower video bitrate from the server. However, when the playback buffer level is almost empty and network conditions are poor, rebuffering events or switches to the lowest video quality can occur [23, 89, 90]. This approach is frequently used, and even Netflix uses a buffer-based approach [21, 25, 97]. However, it exhibits a conservative behavior, starting with a low bitrate and taking a long time to reach the optimal bitrate [91].

The Content-aware rate adaptation algorithm (CARA) buffer-based uses information about the content, packet size in the next selection process, and variance in length of the playout buffer [72]. The process to change the bitrate is carried out in three steps:

- The analysis of the MPD file measures packet throughput at the same time as the network bandwidth.
- An estimation of the expected buffer occupancy with the next chunks avoids buffer underflow and overflow.
- The video bitrate is selected based on the buffering region.

CARA is stable in networks with poor stability conditions and provides high-quality smoothness playout. The stability is based on the content-aware approach with information about the actual packet.

Segment-aware rate adaptation (SARA) [88, 98, 99] is a hybrid/control theory-based algorithm that considers the influence of the actual packet size over the measured bandwidth

to choose an appropriate representation for the next packet. The buffer, which contains the previously downloaded packets, has three Buffer Threshold Values (BTV) based on the packet number. The bitrate changes depend on the buffer fullness and are performed in four phases:

- Fast start: With a buffer level below the first BTV, the lowest VL is selected to reduce the time to display the video content on the client-side.
- Additive increase: With a buffer level greater than the first BTV, the VL only changes gradually, 1 step up or down. This ensures that the buffer fullness does not fall below the first BTV, and it is a conservative approach.
- Aggressive switching: With a buffer level between the first and the third BTV, the bitrate can increase or decrease freely. The idea is to guarantee a gradual and steady increase in video quality.
- Delayed Download: When the buffer level is over the third BTV, requests for new chunks are sent only when the BTV is below the third threshold. In many cases, these steps are unnecessary when the content is paused, and no new content needs to be downloaded.

SARA uses a modified version of the MDP with information about the size of chunks and other details. This solution performs better than a basic adaptation algorithm in low network bandwidth situations.

The Adaptation and Buffer Management Algorithm (ABMA+) [27, 100] employs a pre-computed buffer map (BM) to avoid online computational overload, thereby mitigating issues encountered with overly complex ABR algorithms. ABMA+ predicts the probability of video playout rebuffering for available bitrate representations and selects the maximum bitrate representation that meets a threshold probability. The BM determines the playout buffer size needed to achieve a desired rebuffering probability based on packet download time measured during content download.

Buffer Occupancy based Lyapunov Algorithm (BOLA) was introduced in [21]. It aims to optimize the utility function between average bitrate and rebuffering event duration, taking into account various factors such as content quality, device type, and SSP. BOLA also explicitly defines the relationship between video bitrate and rebuffering probability from the SSP perspective. Although BOLA is effective at reducing rebuffering events, its bitrate selection is deemed overly conservative [101].

2.5.2.1.3. Hybrid algorithms

This approach aims to leverage the main advantages of previous proposals. In [102], the hybrid approach of YouTube that combines server and client-side strategies is described, where ABR bitrates are determined while using a large buffer [103].

One technique at the application layer is zippy pacing [27], which works on the idea of delivering video data just in time. This technique delays the delivery of a chunk until the previous one has been sent. The server sends video chunks with no delay until a specific value of buffer (SVB). Afterwards, the algorithm calculates a pacing delay to maintain the buffer size as close as possible to the SVB. Zippy pacing enables the server to avoid sending

unnecessary chunks, control the target playout buffer, and consider future throughput. Thus, the method allows the server to efficiently control the bandwidth while maintaining the QoE.

In [104], the author proposes an adaptation-aware hybrid client-cache (AAHCC) framework. The proposal considers that, in most research, the client and the cache are independent entities. AAHCC is a hybrid approach that uses a cache pre-fetching scheme to prefetch bitrates. It uses an ABR algorithm based on the forecasted throughput at the cache and client throughput measurements. The framework has a DASH request handler to attend the MPD requests and chunk packets petition from the DASH clients and a cache manager to update/-maintain the requests and prefetch the next chunks to maintain the session. The results show that AAHCC can predict the bitrate for future packets correctly, which reduces the number of unused prefetches.

Pensieve is a proposal which learns ABR algorithms automatically [21, 91]. The system uses observations collected by client video players to train a neural network model that selects bitrates for future video chunks without any explicit rules or assumptions of the operating environment. The approach uses reinforcement learning techniques [21, 91] to develop a series of control policies of both throughputs and buffer through experience. Pensieve learns gradually to make ABR decisions based on the performance of past decisions in the form of reward signals, which helps to adapt to a wide range of environments and QoE specifications.

Reference [105] presents the queuing theory approach to DASH rate adaptation. They model the DASH client as an M/D/1/K queue with three parameters: the packet arrival rate, the service rate, and the capacity of the queue. These parameters implicitly include the playback speed, chunk duration, and buffer size. The packet arrival follows a Poisson distribution, and packets are serviced with a deterministic service rate.

2.5.2.1.4. Model Predictive Control

A hybrid approach known as MPC has been proposed in [106] as a state-of-the-art solution for the QoE optimization problem [107]. The video ABR algorithm is solved by formulating a stochastic optimal control problem using both throughput and VB approaches to achieve a trade-off between them [108]. A sliding look-ahead window is used to predict a series of key parameters, and the obtained prediction is then utilized to solve the optimization problem to achieve higher QoE [108]. The key parameters for QoE are composed of chunk k encoded at bitrate level R denoted by R_k, perceived video quality represented by q, throughput represented by C, the size of chunk k encoded at bitrate level R_k denoted by $d_k(R_k)$, and VB occupancy on the client side represented by B. A more detailed description of these key parameters can be found in [109]. The QoE is expressed using four metrics, namely average video quality (AVQ), average video quality variation (AVQV), rebuffering (Re), and startup delay. The AVQ is the perceived quality q (measured, for example, by PSNR) over all chunks k encoded at bitrate levels R:

$$AVQ = \frac{1}{K}\sum_{k=1}^{K} q(R_k) \tag{2.4}$$

The AVQV is the weighted video quality difference between consecutive chunks:

$$AVQV = \frac{1}{K-1}\sum_{k=1}^{K-1} q(R_{k+1}) - q(R_k) \tag{2.5}$$

The Re is expressed as a combination of the download time of chunk k, $d_k(R_k)$, divided by the throughput C_k, less the VB occupancy B_k:

$$Re = \sum_{k=1}^{K} (\frac{d_k(R_k)}{C_k} - B_k)_+ \tag{2.6}$$

Rebuffering occurs when the download time of a chunk is higher than the occupancy level of the VB on the client side. Finally, the startup delay T_s is the time the user has to wait until the playout, frequently used to fill the VB and test the connection.

The combined QoE metrics for chunk 1 through K are expressed in Equation 2.7, where the weight parameters λ, β, and β_s can be customized based on the relevance of each component. The notation $(x)_+ = max\{x, 0\}$ ensures that the rebuffer parameter in Equation 2.6 is not negative.

$$QoE_1^k = AVQ - \lambda * AVQV(K-1) - \beta * Re - \beta_s * T_s \tag{2.7}$$

The MPC approach has demonstrated its effectiveness in improving user QoE by taking into account both video quality and rebuffering behavior in video streaming. In summary, MPC is a QoE optimization solution for video streaming that considers the trade-off between throughput and buffer approaches by predicting variables over a look-ahead horizon. The approach defines four key parameters, and it formulates a stochastic optimal control problem that minimizes the combined metrics for chunks 1 through K. By considering the importance of video quality and rebuffering behavior, MPC has proven to be an effective solution to enhance user QoE in video streaming.

3. Methodology

The methodology chapter elaborates on the development and implementation of the ASViS protocol, aimed at optimizing video transmission over fluctuating network conditions. It introduces an advanced algorithm at the application layer, which employs a layer-discarding policy and deadline-sensitive criteria for dynamic adaptation to network changes, ensuring prioritized data transmission. This approach, underpinned by the use of SVC and UDP for data transmission, addresses the first specific research objective by demonstrating a robust and effective adaptive video transmission solution. The modeling of ASViS, targeting the second specific objective, showcases its capability to adapt dynamically and maintain video quality, even with limited bandwidth. Optimization and evaluation of ASViS, critical for the third specific objective, involve detailed analysis and a multidimensional method, highlighting ASViS efficiency over traditional ABR protocols. This comprehensive methodology ensures not only improved video transmission efficiency under variable network conditions but also a consistent, high-quality user experience, aligning the experiments with the research objectives to contribute meaningfully to the field of adaptive video streaming.

3.1. Connection between Methodology and Experiments with Specific Objectives

The methodology adopted in this study focuses on the design and implementation of the ASViS protocol, an inter-layer solution designed to optimize adaptive scalable video transmission under fluctuating network conditions. ASViS implements an advanced algorithm at the application layer that manages the sent information, based on a layer discard policy and deadline-sensitive criteria. This innovative approach allows for dynamic adaptation to changing network conditions, prioritizing data transmission based on the importance of video layers and time constraints.

- Design and Implementation of ASViS: The design and implementation process of ASViS, SVC and UDP for data transmission, is fundamental to achieving the first specific objective of this research. The configuration of the 'flow window' or 'fwnd', similar to TCP congestion window, along with the implementation of a Selective Acknowledgment (SACK) mechanism for Round-Trip Time (RTT) estimation and available bandwidth calculation, demonstrates our ability to develop a robust and effective solution for adaptive video transmission.

- Modeling of ASViS: The modeling of ASViS directly addresses the second specific objective. This model manages packet discards when bandwidth is limited, implementing flow control at the application layer and mapping video data to packet data. The simulation of video transmission under various network conditions validated the efficacy of ASViS, highlighting its ability to dynamically adapt and maintain video quality.

- Optimization and Evaluation of ASViS Configuration: Optimizing ASViS parameters, including the τ setting to manage curfew gaps between layers, is essential for meeting

the third specific objective. This optimization was conducted through detailed analysis and a multidimensional method, ensuring the adaptability and efficiency of the protocol under variable network conditions. The evaluation of these configurations through the experiments described in Chapter 5 demonstrated the superiority of ASViS over conventional ABR protocols, providing a consistent and high-quality user experience.

Each stage of the adopted methodology has been carefully designed to address the specific objectives of this research, from the development of the protocol to its optimization and evaluation. This comprehensive approach ensures that ASViS not only improves the efficiency of video transmission under fluctuating network conditions but also provides a consistent and high-quality user experience. Aligning our experiments with these specific objectives reinforces the validity and relevance of our findings, significantly contributing to the field of adaptive video streaming.

3.2. ASViS Overview

ASViS, or adaptive scalable video streaming, is a cross-layer solution that bridges the gap between the transport and application layers. Within the application layer, an algorithm is implemented that manages the dispatched information, relying on a layer-discarding policy and deadline-sensitive criteria. While packet loss assists in determining the flow rate, reflecting the behavior of TCP, retransmissions are not essential. Based on the layer-discarding policy, certain packets might be omitted. Utilizing UDP, the system detects missing frame fragments at the application layer. Since there is no requirement to track sequence numbers or acknowledge every packet, UDP becomes a preferred choice. The flow rate is managed by adjusting the number of packets dispatched over a time span. This approach, similar to the congestion window of TCP, is termed here as the flow window or fwnd. ASViS supports SACK; however, it is dispatched only when deemed essential. SACK packets are vital for measuring RTT, which is required for estimating available throughput. Lost packets might be retransmitted if the time and priority permit. The flow window is principally employed to fulfill with RFC 8085 requirements which emphasize preventing network saturation [2].

Scalability is an important factor in the growth of any technology. When referring to ASViS, scalability is characterized by several factors: 1. Since ASViS is based on SVC, which itself is part of H.264 (and newer), it can easily evolve to higher resolutions (8K, 16K, etc.). 2. SVC does not have a fixed amount of layers, to prove our concept this work uses four SVC layers, but more can be used. As technology progresses and resolutions increase, it is beneficial to also increase the amount of SVC layers used, since it would create more granularity (i.e., more quality options to choose from). There are other aspects not covered in this work, in which ASViS can scale (such as GOP size, block compression size, etc.) that are inherent to the compression standards. All these degrees of scalability allow ASViS to maintain itself relevant in the future.

Compared to traditional video streaming methods based on TCP, which often rely heavily on retransmissions and sequence acknowledgments, ASViS, with its refined layer-discarding policy and strategic use of UDP, offers a more efficient and adaptable approach. This methodology not only reduces overhead but also aligns closely with the RFC 8085 guidelines. These guidelines are a series of recommendations on UDP usage and best practices designed to prevent network congestion. By complying with these guidelines, stability and efficiency in

network operations are ensured. The adherence of ASViS to this standard is underscored by its significance.

A deadline-sensitive approach is employed, which operates on a time-based metric. This metric is focused on the residual time to meet the playout frame deadline. In SVC codification, by default, all layers of a frame share the same deadline. This approach introduces differentiation, suggesting the omission of certain layers based on pre-set criteria using the deadline as a reference. A significant challenge in video streaming emerges when throughput is below the video bitrate. Stream prioritization becomes essential, with low-priority data being discarded as deemed necessary. The inherent structure of SVC layers, where higher layers rely on foundational ones, designates a higher priority to base layers. As the deadline becomes imminent, high-layer data are omitted, and if the situation demands, additional layers are also removed.

The proposed layer-discarding policy suggests having differentiated deadlines for distinct layers. To distinguish it from the frame deadline, the layer deadline is referred to as *curfew*. Higher layers (lower priority) have earlier curfews, and lower layers (higher priority) have more lenient curfews. The curfew spacing is an important design parameter. The algorithm may accept a low-priority layer at the cost of running out of time to process a subsequent higher-priority layer if a late curfew (short present-to-deadline period) is used. However, if the curfew is too early, the algorithm response may be premature, and higher-layer (enhancement) packets can be unnecessarily discarded. The decision-making process is a critical part of the proposed solution. It differentiates discarding criteria to specific layers for an SVC video with two layers (BI, BB, EI, and EB), as shown in Figure 3.1 where all layers of a frame have a defined threshold.

- Data received after the deadline are discarded, regardless of their priority.
- Only BI layers are considered for data that arrives between the deadline and the first curfew.
- At the second threshold, both BI and BB layers are accepted.
- When the third threshold is reached, BI, BB, and EI layers are permitted.
- Beyond the third threshold, all layers, inclusive of EB, are accepted.

Deadline-sensitive Criteria Algorithm (DSCA). The decision-making process, shown in Figure 3.2, consists of reading a packet and extracting the frame (I or B) and layer (base or enhancement) information. Based on this it computes the curfew, which considers the current time, frame deadline, travel time, and priority. The curfew is computed only on packets that are inside the current *fwnd*, as later packets depend on current decisions. Higher priority packets have curfews closer to the deadline, while lower priority packets are discarded earlier to increase the chances of a higher priority packet arriving on time. If a packet inside the current *fwnd* is discarded, a later packet is pushed to an earlier time increasing its probability of successfully arriving before the deadline. This is the fundamental principle, increasing the chances of successfully arriving before the deadline.

In the realm of digital video streaming, the importance of scalability and future-proofing for protocols such as ASViS is underscored by the rapid advancements in video resolution and network technology. It is recognized that ASViS is designed to adapt seamlessly to these advancements, owing to its foundational reliance on Scalable Video Coding (SVC). As

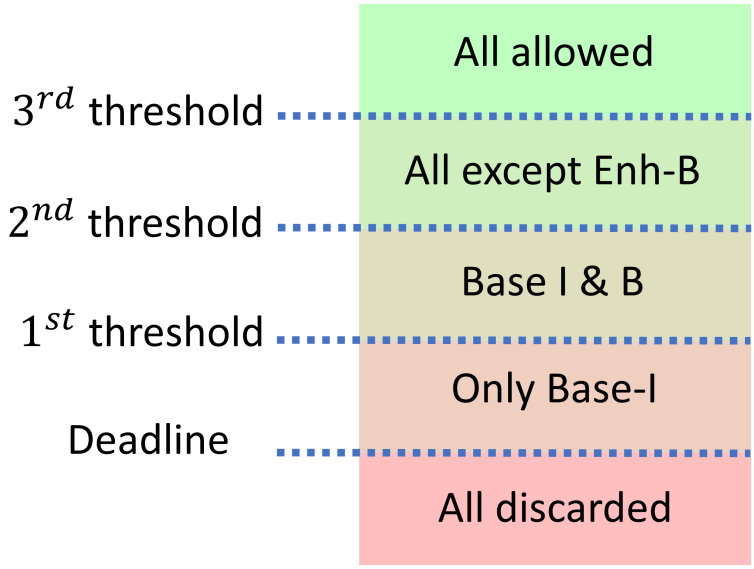


Figure 3.1: Layers and discarding criteria.

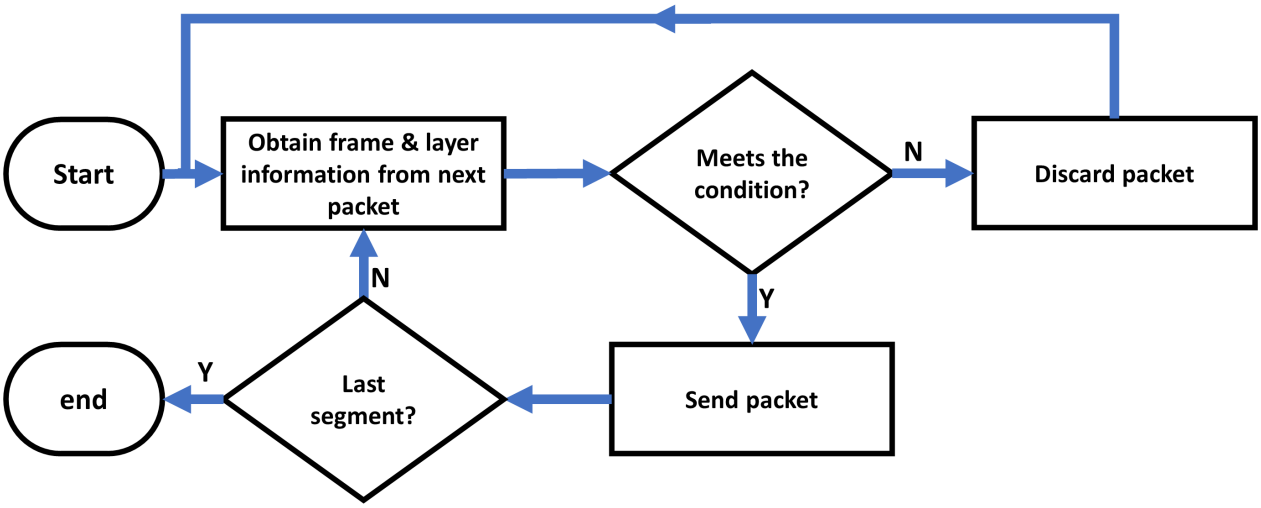


Figure 3.2: Packet selection process flowchart of DSCA.

resolutions progress towards higher standards, including 8K and beyond, the architecture of ASViS facilitates the integration of such advancements. This integration is achieved through the flexible nature of SVC layer structure, which is utilized by ASViS for effective video stream management. The protocol ability to incorporate an increasing number of SVC layers enhances its granularity in video quality, thereby ensuring compatibility with future video codecs and resolutions.

When the implications of implementing ASViS are considered, particularly in terms of cost, scalability, and user experience, it is observed that the protocol could significantly impact video streaming services and their consumers. The cost-effectiveness of ASViS is attributed to its efficient use of network resources, while its scalability ensures that it remains relevant in the face of evolving video technologies. From the perspective of user experience, ASViS promises to enhance the quality of video streaming, adapting to varying network conditions without significant degradation in video quality.

Moreover, the efficiency of ASViS, especially regarding network resource utilization and responsiveness to changing network conditions, is a focal point of its design. The algorithm is structured to minimize unnecessary network load, thereby optimizing the consumption of bandwidth. In response to fluctuations in network conditions, ASViS demonstrates a high degree of adaptability, adjusting packet management strategies in real-time to maintain optimal video streaming quality. This responsiveness not only conserves network resources but also ensures a consistent and high-quality user experience, regardless of the variability in network performance.

3.2.1. Modeling ASViS

The quality of received video is contingent on available bandwidth. ASViS is designed to manage packet discards when bandwidth is limited. At the application layer, flow control is implemented, similar to TCP behavior. However, unlike TCP, ASViS allows the sender to decide on packet discards, which is why UDP is utilized. The receiver responds with SACK, enabling the sender to adjust the flow upon detecting packet losses.

The initial step is to map the video data to the packet data. The simulation assumes that each video frame is sent in its own assigned packet or several packets if the video frame is larger than the maximum segment size (MSS). Nevertheless, two video frames are not encapsulated in a single packet to avoid losing multiple frames with a single packet loss. To avoid confusion, the term *frame* is strictly used in the sense of a video image, not a data-link layer unit.

The bitstream of the video is given by $b_S = \frac{d_T}{t_T}$, where d_T is the total data to be transmitted, including all the layers, and t_T is the total encoding time. The total data can be expressed as:

$$d_T = \sum_{f=1}^{F_T} \sum_{l=1}^{L} d_{\langle f,l \rangle}. \tag{3.1}$$

F_T is the total amount of frames and L is the total amount of layers per frame. $d_{\langle f,l \rangle}$ is the amount of data needed to transmit the l^{th} layer of the f^{th} frame. The bitstream is varied according to the available bandwidth, considering discarded packets, so the effective bitstream b_F is:

$$b_F = \frac{d_F}{t_T} \text{ and } d_F = \sum_{f=1}^{F_T} \sum_{l=1}^{l_F} d_{\langle f,l \rangle}. \tag{3.2}$$

The value d_F reflects the total data successfully transmitted, accounting for the layer-discarding decisions made based on ASViS flow control policies. l_F is the number of layers transmitted for frame f, where l_F could have a value of 0 if the whole frame is discarded. Since the transport layer is flow-controlled it is necessary to determine the number of packets required. The flow can be controlled on a packet or byte basis. In most cases, protocols operate on a packet basis. The amount of packets necessary to transmit all the frames that are not discarded (arrive successfully) is:

$$S_{\langle f,l \rangle} = \left\lceil \frac{d_{\langle f,l \rangle}}{MSS} \right\rceil. \tag{3.3}$$

To simplify the frame and layer indexing to a single indexing variable the following conversion can be made: $q = L\,(f-1) + l$, where L is the total amount of layers, l is the layer index per frame spanning from 1 to L, and f is the frame index varying from 1 to F_T. Therefore, q spans from 1 to $L\,F_T$, and (3) becomes $S_q = \lceil d_q/MSS \rceil$. The *fwnd* varies with each RTT cycle depending on whether packets arrive successfully at the client-side. The *fwnd* is represented by a vector where each entry corresponds to an RTT cycle w, where W is the total number of cycles, i.e., $\overrightarrow{fwnd} = \{fwnd_1, \ldots, fwnd_w, \ldots, fwnd_W\}; w \in \mathbb{N}$.

Given the packet q, the function C_q outputs the RTT cycle w where the last packet of layer q is found. For better accuracy, a packet-acceptance pattern should be included, which is a set of ones and zeros representing true and false statements, respectfully, responding to whether the packet successfully arrived at the destination and before the deadline. This is represented by the vector $\overrightarrow{X} \in \{0, 1\}$. There is a correlation between the number of ones in $\overrightarrow{X}$ and the overall quality of the video.

To model the behavior of the packet-acceptance pattern, it is necessary to perform an iterative computation to determine if that layer is to be discarded or retained. The **first step** involves determining the deadline and curfew of the layer. To determine this the RTT cycle is needed, and given by C_q:

$$C_q = \arg\min_{w} \left\{ \sum_{n=1}^{w} fwnd_n - \sum_{\hat{q}=1}^{q} S_{\hat{q}} X_{\hat{q}} > 0 \right\}. \tag{3.4}$$

S_q is the number of packets required to send layer q. X_q is the q^{th} value of the packet-acceptance pattern vector, but for this stage we assume $X_q = 1$. All values earlier than X_{q-1} (inclusive) are known.

Second step. With the RTT cycle C_q information, the arrival time of the packet q (or t_{A_q}) is computed as:

$$t_{A_q} = RTT \cdot \left(C_q \Big|_{X_q=1} - 1/2 \right). \tag{3.5}$$

Third step. The deadline of frame f (or D_f) is a straight-forward computation obtained using the buffer time t_B and the frame rate r, hence $D_f = \frac{f}{r} + t_B$. It is beneficial to obtain a deadline expression in terms of the packet index q (or D_q). Since all layers of a frame have the same deadline the relation $f = \lfloor q/L \rfloor$ is sufficient to compute D_q. Substituting gives:

$$D_q = \left\lfloor \frac{q}{L} \right\rfloor \frac{1}{r} + t_B, \tag{3.6}$$

Fourth step. To retrieve the priority of packet q (or p_q) for P levels of priority, it is necessary to know the GOP pattern. As mentioned earlier and shown in Fig.2.5, the GOP pattern is IBBBBBBB. To obtain the priority level of this pattern as a function of layer q the following expression is used:

$$p_q = \begin{cases} 2(q-1) & \text{if } q \le L \\ 2\,\Pi(L) & \text{if } q > L \;\&\; \Pi(L\,G) < L \\ 2\,\Pi(L)+1 & \text{if } q > L \;\&\; \Pi(L\,G) \ge L \end{cases}, \tag{3.7}$$

where $\Pi(\lambda) = mod(q - L - 1, \lambda)$ and G is the GOP size, in this case, $G = 8$. p_q spans from 0 to $P-1$, where the lower values have higher priorities. Note: q is in the coding order

(see Fig.2.5).

Fifth step. Once the priority is found, the curfew Γ_q is determined on Equation 3.8, where the number of priority levels is obtained by $P = L\,T$ where T is the number of frame types. Here, we work with I and B frames, so $T = 2$.

$$\Gamma_q = mod(p_q, P)\cdot\tau \tag{3.8}$$

τ is a positive value in units of RTT that determines the time spacing between priority levels, it is a design parameter. The greater the value the more conservative the algorithm, and the lower is riskier. Increasing τ increases the chances of having a smooth video stream, lowering τ attempts to transmit a higher quality (more layers) stream but with a higher probability of loss due to limited bandwidth.

Final iteration step. The packet-acceptance pattern set can be obtained as the transmission conditions are met using:

$$X_q = \begin{cases} 1 & \text{if } D_q - t_{A_q} - \Gamma_q \geq 0 \\ 0 & \text{if otherwise} \end{cases}. \tag{3.9}$$

The value of X_q, which at the beginning of the iteration was assumed to be 1, is now determined. This value is used for the subsequent iteration. A visual representation of the iterative process of the proposed solution, ASViS, for each one of the packets that compose a video frame, also can be seen on Figure 3.3, where we can see how we need the information of both transport and application layer to take the decision.

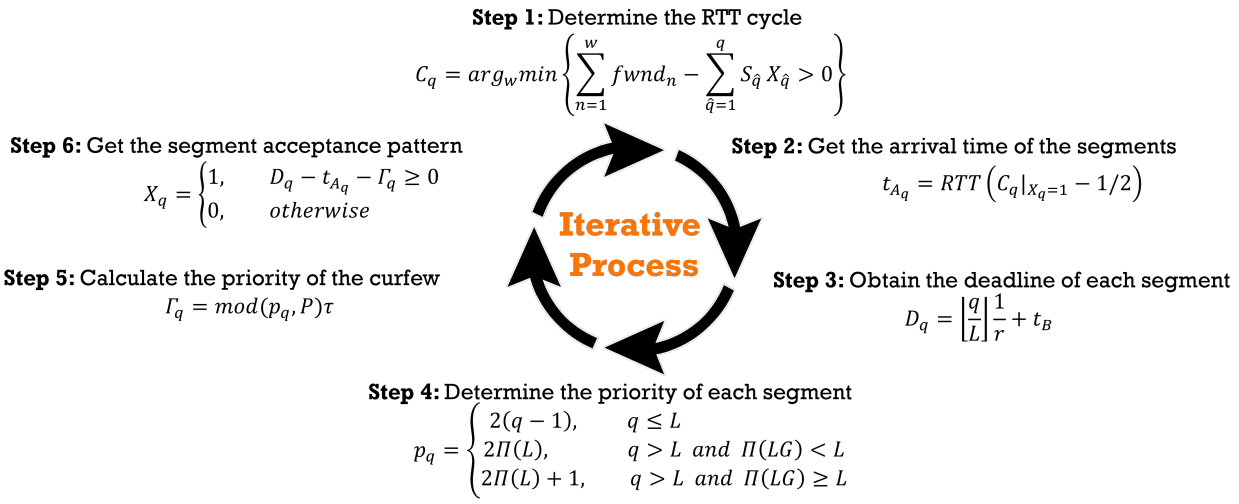


Figure 3.3: Iterative process of ASViS to send or discard a packet.

In the modeling of ASViS, emphasis is placed on the dynamic optimization of video quality, adapting to available bandwidth. Adaptability is achieved through a sophisticated decision-making process, considering the urgency and importance of each packet based on frame deadlines and layer priorities. A selective acknowledgment strategy is implemented, enabling effective assessment of network conditions and adjustment of transmission strategies. This approach ensures optimal utilization of network resources, highlighting the models adaptability and efficiency under diverse network conditions.

In the model, a comprehensive packet-acceptance pattern is incorporated, determined

through iterative computations. This pattern plays a crucial role in deciding whether specific layers of video data should be transmitted or discarded. Decisions are continuously revised based on real-time analysis of network conditions and packet reception success rates. Furthermore, video frames are managed in distinct packets without overlap, minimizing the risk of compounded frame losses. This methodical packetization aligns with the protocols objective of maintaining high video quality, even in challenging network scenarios.

3.2.2. Optimization Method for τ Configuration

In the study presented in [110], a proof-of-concept is demonstrated in which video quality is maintained without re-buffering. However, an in-depth formal analysis for optimizing the parameter τ, representing curfew gaps, is not provided. An exploration into varying τ values within a reasonable range reveals that the maximum Y-PSNR is indicative of optimal video quality.

For optimization scenarios aiming at a single objective such as a root, maximum, or minimum, the bisection method is commonly employed [111, 112]. However, its limitation in potentially identifying local maxima instead of the global maximum is acknowledged. Therefore, a multi-section variant of the bisection method is utilized in this study.

A multi-dimensional multi-section (MM) method is introduced, extending the bisection method for multi-dimensional optimization while minimizing the likelihood of identifying local maxima. Unlike traditional bisection methods, which divide the search space into two [111, 112], a more expansive search technique is adopted here to mitigate the risk of mistaking local maxima for global ones.

Various applications show a monotonic increase up to a maximum, followed by a monotonic decrease, making traditional methods effective. However, for systems exhibiting oscillatory magnitudes, a more cautious approach is warranted. In this study, multiple points per dimension are probed in each iteration, with the step size and search region being adjusted based on the identified maximum. Achieving satisfactory result precision within a reasonable computational time is a non-trivial task, particularly due to the discrete nature of the data. The algorithmic details of the Multi-probe Bisection Method are presented in Algorithm 1.

Algorithm 1 multi-dimensional Multi-section Method (3D example)

$[m_1, m_2, m_3] \leftarrow [0, 0, 0]$
$[M_1, M_2, M_3] \leftarrow$ upper limit of τ_1, τ_2, τ_3
$range_x \leftarrow (M_x + m_x)/2$ **for** $x = 1, 2, 3$
$N \leftarrow$ *number of steps*
$step_x \leftarrow range_x/(N-1)$ **for** $x = 1, 2, 3$
$Q \leftarrow$ *empty matrix* (1×0)
$list_{\tau_1,\tau_2,\tau_3} \leftarrow$ *empty matrix* (3×0)
while error tolerance (init $i \leftarrow 0$ before loop)
  for τ_1 **from** m_1 **to** M_1 **step** $step_1$
    for τ_2 **from** m_2 **to** M_2 **step** $step_2$
      for τ_3 **from** m_3 **to** M_3 **step** $step_3$
  if $[\tau_1, \tau_2, \tau_3]$ **is found in** $list_{\tau_1,\tau_2,\tau_3}$ **then** *skip*
  else (increase $i \leftarrow i + 1$)
    add $[\tau_1, \tau_2, \tau_3]$ **to** $list_{\tau_1,\tau_2,\tau_3}$ **in row** i
    run test of video quality
    add quality result **to** Q **in row** i
  end if
      end for
    end for
  end for
  $range_x \leftarrow range_x/2$ **for** $x = 1, 2, 3$
  $step_x \leftarrow step_x/2$ **for** $x = 1, 2, 3$
  $Q_{MAX} \leftarrow \max(Q)$
  $i_{MAX} \leftarrow$ indices of Q where $Q = Q_{MAX}$
  $list^{MAX}_{\tau_1,\tau_2,\tau_3}$ **extract rows** i_{MAX} **from** $list_{\tau_1,\tau_2,\tau_3}$
  $m_x \leftarrow \min(\tau_x) - range_x$ **for** $x = 1, 2, 3$
    where τ_x **belongs to** $list^{MAX}_{\tau_1,\tau_2,\tau_3}$
  $M_x \leftarrow \max(\tau_x) + range_x$ **for** $x = 1, 2, 3$
    where τ_x **belongs to** $list^{MAX}_{\tau_1,\tau_2,\tau_3}$
end while

4. Experimental Setups and Results

Experiments were conducted in a structured manner to evaluate the performance of the ASViS algorithm in comparison to conventional methods. Each investigation was targeted towards elucidating specific attributes and behaviors. The performance of ASViS is influenced by several specific parameters, which will be elaborated upon in the subsequent sections. Notably, network conditions, such as RTT, packet loss, and buffer conditions, have been identified. Additionally, variations in τ parameters across different layers and alterations in the size of each layer have been observed to impact performance. A thorough examination of these factors and their implications on ASViS functionality is included in the following discussions. The first experiment juxtaposed a theoretical model of ASViS, particularly in assessing the anticipated arrival time of frames at the client side, against an empirical counterpart. The purpose was to determine the accuracy of the theoretical model in simulating network dynamics. The second experiment delved into the perceived video quality by modulating the parameter τ. By setting a continuum of equidistant benchmarks, the Y-PSNR was gauged, revealing a complex interrelation between the τ values and the Y-PSNR attained. In the third analysis, the objective was to discern the optimal curfew gaps, represented as the set of taus, suitable for static initial network scenarios. Here, the MM method paved the way to unearthing superior video quality. Unlike assuming uniform curfew gaps, this study hypothesized better outcomes from distinct-sized intervals, with the Y-PSNR being the primary metric of evaluation. The fourth exploration sought to validate the theory that layer-based deadline gaps echo the proportions of the layer sizes. The culminating experiment instituted a face-off between two ABR protocols, ASViS and MPC. Using a spectrum of metrics, from VB behavior and throughput to video quality, this comparative study decoded the operational nuances of ASViS juxtaposed with the MPC algorithm.

The video utilized for simulations comes from Xiph video repository, specifically the 352x288 .yuv file, encoded at 30 fps. The JSVM facilitated video coding and decoding on SVC. Video details and parameters are provided in Table 4.1. Insights into parameters of coded video files are provided by the JSVM trace file, which facilitates video reconstruction using specific layers or packets. The ASViS algorithm was implemented using MATLAB and JSVM. Different network conditions were simulated based on the layer information extracted from the JSVM trace file, determining whether packets were transmitted or discarded. Furthermore, MATLAB was employed in conjunction with JSVM for tasks associated with video encoding/decoding. Additionally, the results were organized and processed using MATLAB. JSVM works using the operating system command prompt. It is easy to link JSVM to MATLAB using the matlab *system* command. The video quality of different ABR algorithms was gauged using PSNR and for VMAF testing, version 0.6.1 was employed from github, incorporating default pre-trained machine learning coefficients.

4.1. Results

The objective of the first experiment is to compare the network behavior outcomes of a theoretical ASViS model with those of an experimental test. Both scenarios employ UDP

Table 4.1: Video parameters for all experiments.

Parameter	Details
File	$bus\_cif.y4m$
Rate	30 fps
Length	150 frames
Number of layers	2
Quantization Layer 1 - BL	40
Quantization Layer 2 - EL	20
GOP	8
Resolution	width 352, height 288
Amount of BI/EI	19
Amount of BBI/EB	121
Y-PSNR (Orig. Encoded)	40.62 [dB]
VMAF (Orig. Encoded)	99.97 [%]

transmission rules equipped with flow control to prevent congestion. In the theoretical model, the determination of which packets are discarded is based on the estimated packet size (EPS) and network conditions as utilized by the ASViS algorithm. The EPS values are informed by the mean layer size information derived from a trace file, which is created for JSVM. The size in MSS for each layer is presented in Table 4.2. For the experimental model, the actual size of each layer, influenced by its content, whether it is BL or EL, and its classification as I or B, is utilized. This results in a variety of sizes for each layer. The size in $[kB]$ for each layer is presented in Table 4.2. Both models operate with the same network transmission parameters: RTT 0.05 s, packet loss rate 0.01, and buffer 3 s. Results are calculated based on 100 pseudo-random seeds.

Table 4.2: EPS for each layer for ASViS theoretical experiment.

Layer	EPS $[kB]$
BI	7154
BB	584
EI	36500
EB	11388

Within the ASViS model, the behavior can shift from conservative to risky by adjusting the τ values. For a video comprising four layers, it might be presumed that there would be four distinct τ values representing BI, BB, EI, and EB. However, in this experiment, the τ value for the BI layer is set at 0 due to its importance. This ensures its transmission until the deadline is reached. Seven configurations, each with an equidistant and incremental τ gap (τG) based on layer priorities, are outlined in Table 4.3.

The results, both theoretical and experimental, appear in Figure 4.1. This representation showcases the estimated arrival time of each frame at the client end. The gradient of the slope conveys the transmission rate; a gentler slope indicates a higher rate, whereas a steeper

Table 4.3: 7 τ gap configurations to ASViS.

τG	τ_2 - BB [RTT]	τ_3 - EI [RTT]	τ_4 - EB [RTT]
τG_1	1	2	3
τG_2	2	4	6
τG_3	3	6	9
τG_4	4	8	12
τG_5	5	10	15
τG_6	6	12	18
τG_7	7	14	21

RTT of 0.05 s.

one suggests a slower rate. A significant gap between theoretical and experimental results is observed for τG_7. This discrepancy can be attributed to various factors, including differences in average vs. actual packet size and video complexity variations.

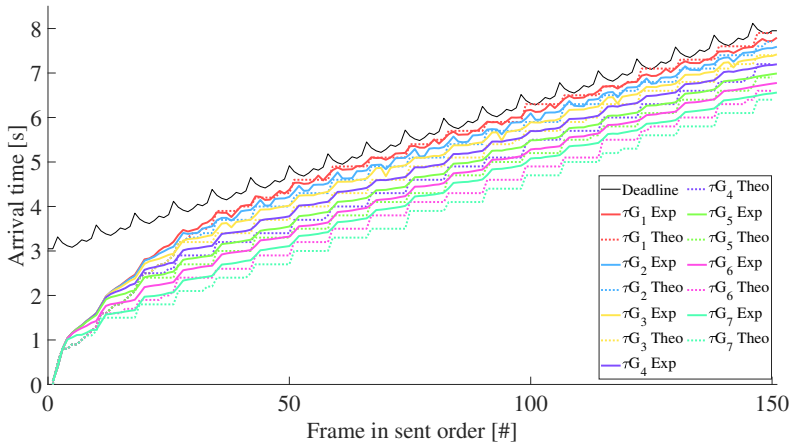


Figure 4.1: Results of the arrival time of frames for theoretical and experimental scenarios.

To determine the accuracy of the correlation between models, the mean absolute percentage error (MAPE) is employed. Table 4.4 provides a detailed breakdown, pointing out a maximum prediction error of approximately 14.2% for τG_6. Given the inherent variability of the practical experiments due to packet losses and the intricate characteristics of each frame, along with the nuances of the various layers and complexities associated with network conditions in empirical analyses, the accuracy of the theoretical model is underscored. It is noteworthy that high fidelity is achieved by the theoretical model, aligning well with the experimental results. This contributes to a reduction in the need for exhaustive and computationally demanding empirical tests, thereby facilitating more streamlined behavioral predictions based on the theoretical model. It is important to stress that this does not replace empirical benchmarks, but can significantly reduce the amount of energy, resources, and time devoted to more practical tests.

The objective of the second experiment is to examine the impact of varying tau values

Table 4.4: MAPE comparison from theoretical and experimental results of ASViS.

τG	MAPE [%]
τG_1	6.5
τG_2	7.3
τG_3	9.3
τG_4	10.7
τG_5	10.1
τG_6	14.2
τG_7	13.7

on the video quality performance of ASViS, focusing specifically on the experimental results. For this purpose, two network scenarios are considered. Scenario 01 operates with a RTT of 0.05 seconds, a packet loss rate of 0.01, and a 1-second buffer. Scenario 02 is characterized by an RTT of 0.1 seconds, a packet loss rate of 0.01, and a 3-second buffer. Both scenarios are analyzed using 100 pseudo-random seeds for each tau value. The configuration of τG values follows the format specified in Table 4.3. Additionally, this study evaluates the behavior of SVC in scenarios without ASViS, referred to as the 'no protocol' condition.

Figure 4.2, depicts the scenario 01 with a shorter round-trip time (RTT) of 0.05 seconds and a streamlined 1-second buffer, which manifests a notable punctuality in frame arrival times across the various τG configurations, aligning more closely with the deadline frame. This scenario illustrates the protocol proficiency in a more constrained environment where timely data throughput is paramount. Conversely, Figure 4.3, denoted as Scenario 02, illustrates a contrasting setting with an RTT of 0.1 seconds coupled with a 3-second buffer. Here, despite the inherent latency challenges, the ASViS protocol demonstrates a robust capacity for maintaining video streaming efficiency, as evidenced by the frame arrival times, which consistently surpass the 'no protocol' scenario, albeit with a slightly more staggered pattern due to the increased buffer size. The comparative analysis of both scenarios elucidates the adaptive strength of ASViS, affirming its capability to enhance the streaming experience by dynamically adjusting to varied network conditions while substantially outperforming the 'no protocol' case in terms of consistent and timely frame delivery.

Figure 4.4 and Figure 4.5 illustrate the impact of network conditions on video quality within the ASViS protocol. Contrary to initial expectations, Scenario 2, as shown in Figure 4.5, with its higher RTT and larger buffer, demonstrates a marginal advantage in the 'no protocol' baseline with a Y-PSNR of 37.3 dB compared to scenario 1 at 35.1 dB, suggesting that increased buffering may mitigate packet loss and enhance quality when ASViS is not in use. However, under the ASViS regime, scenario 1 shown in Figure 4.4 consistently outperforms scenario 2 across all τG levels, with Y-PSNR readings declining from 39.7 dB at τG_1 to 38.8 dB at τG_7, indicating that the protocol effectively leverages tighter network constraints to maintain higher video quality. In contrast, scenario 2 experiences a steeper descent in quality, descending to 35.5 dB at τG_7, highlighting the relative sensitivity of ASViS to increased RTT and buffer conditions, this data underscores the protocol adeptness in fine-tuning video transmission, particularly in less buffered environments, while also revealing room for optimization in scenarios with more significant latency.

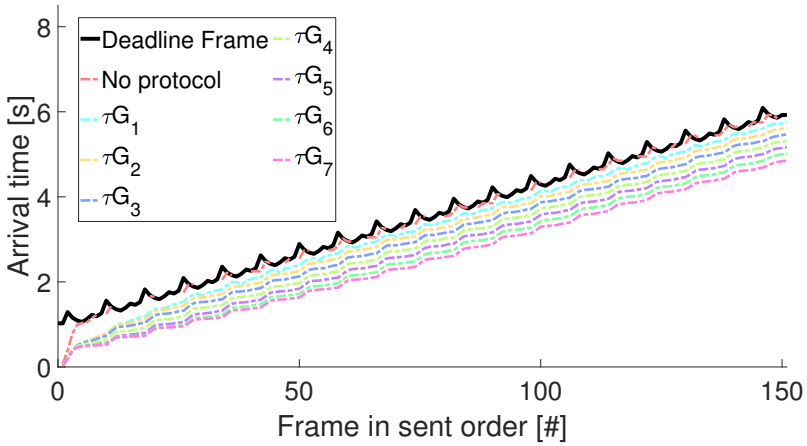


Figure 4.2: Arrival time of frames across varying τG settings compared to the 'no protocol' condition in Scenario 01, with RTT of 0.05 seconds, packet loss rate of 0.01, and a 1-second buffer.

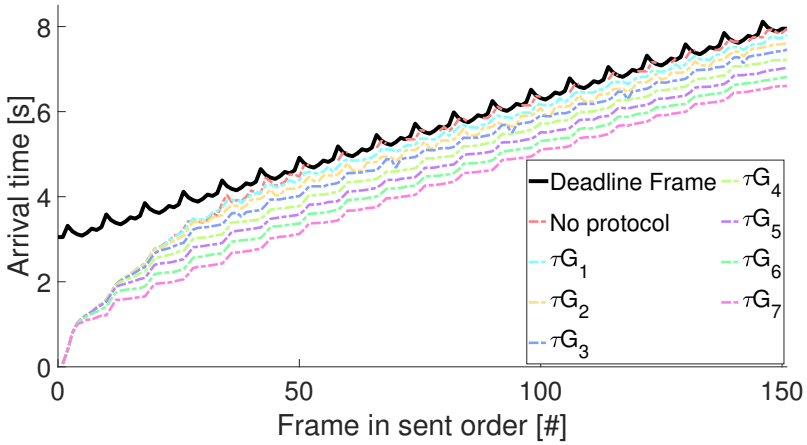


Figure 4.3: Arrival time of frames across varying τG settings compared to the 'no protocol' condition in Scenario 02, with RTT of 0.1 seconds, packet loss rate of 0.01, and a 3-second buffer.

Analysis of video playout buffer stability across scenarios 1 and 2 is captured in Figure 4.6 and Figure 4.7, underscoring the effectiveness of the ASViS protocol. In Scenario 1, shown in Figure 4.6, ASViS consistently maintains buffer size well above the level observed in the absence of the protocol, with the τG_7 configuration demonstrating an exemplary buffer size at full video playout. This robust performance in a low-latency environment reflects the protocol proficient buffer management capabilities. Figure 4.7 illustrates that in the higher latency environment of Scenario 2, ASViS significantly outperforms the 'no protocol' scenario, which suffers from complete buffer depletion after only a quarter of video playback. With ASViS, the ascending trend in buffer size across increasing τG values signifies the protocol strong adaptive response, effectively circumventing the potential for rebuffering incidents. These observations corroborate the pivotal role of ASViS in ensuring a smooth streaming experience, dynamically optimizing buffer capacity to address varying network conditions.

In the third experiment, the objective is to pinpoint optimal τ values to maximize specific

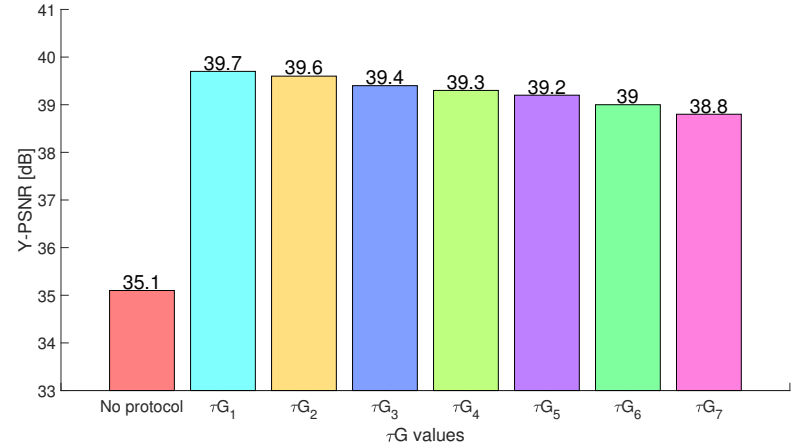


Figure 4.4: Y-PSNR performance for G and no protocol condition in Scenario 01.

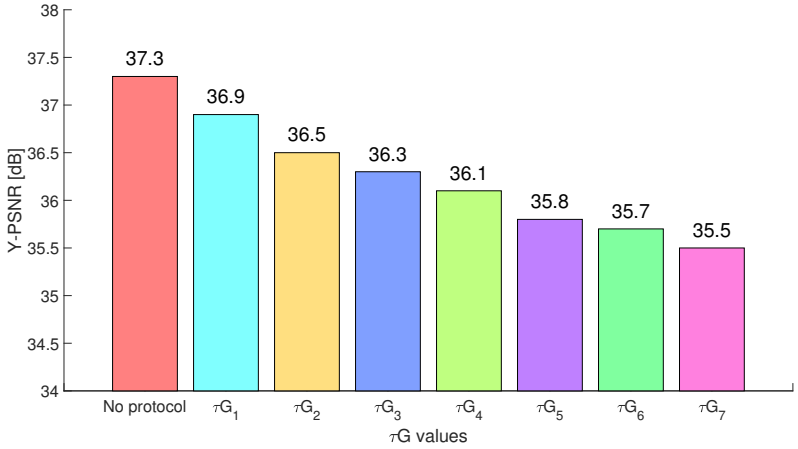


Figure 4.5: Y-PSNR performance for G and no protocol condition in Scenario 02.

criteria. For this experiment, the optimization objective is the performance of ASViS, based on the average amount of sent layer outcomes, using the MM method. It is crucial to note that these optimal values might not always be integers, suggesting a broad range of potential τ values. The foundational setup for this experiment involved fixed parameters such as an RTT of 0.05 s, a packet loss rate of 1/200, and a buffer of 1 s. For BI, a fixed value of τ_1 at 0 seconds (representing the deadline) was applied. In contrast, for the layers BB, EI, and EB, the values τ_2, τ_3, and τ_4 were adjusted, respectively. Performance was evaluated based on the maximum number of layers transmitted, with a combined limit of 300 layers: 19 for BI, 19 for EI, 141 for BB, and 141 for EB. This evaluation determined the local maxima within each quadrant. For each point, ten pseudo-random seeds were probed, and the average of the maximums was obtained. A consistent search range, spanning from 0 to 8 seconds, was employed for each τ.

The visualization of these results is provided in a 3D plot, as seen in Figure 4.8. The axes, labeled as G_2, G_3, and G_4, represent the gaps (G) or intervals between consecutive τ values, excluding τ_1. These represent the intervals from the deadline to τ_2, from τ_2 to τ_3, and

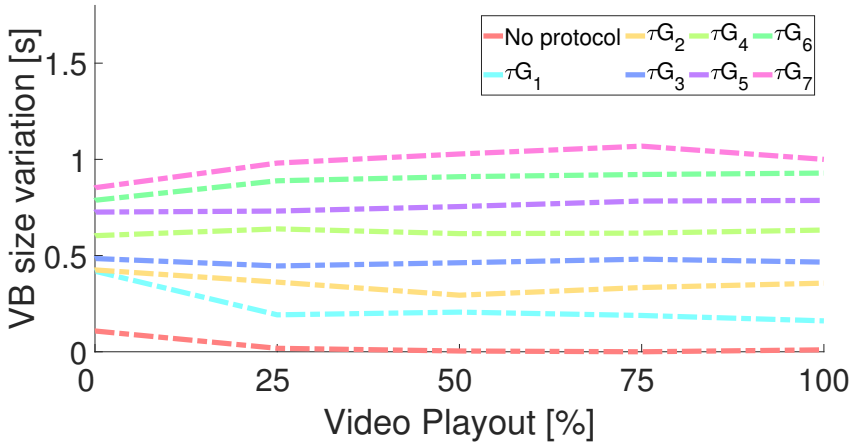


Figure 4.6: Variations in video buffer size relative to playout percentage for Scenario 01, illustrating the influence of ASViS under network conditions with RTT of 0.05 seconds, packet loss rate of 0.01, and a 1-second buffer.

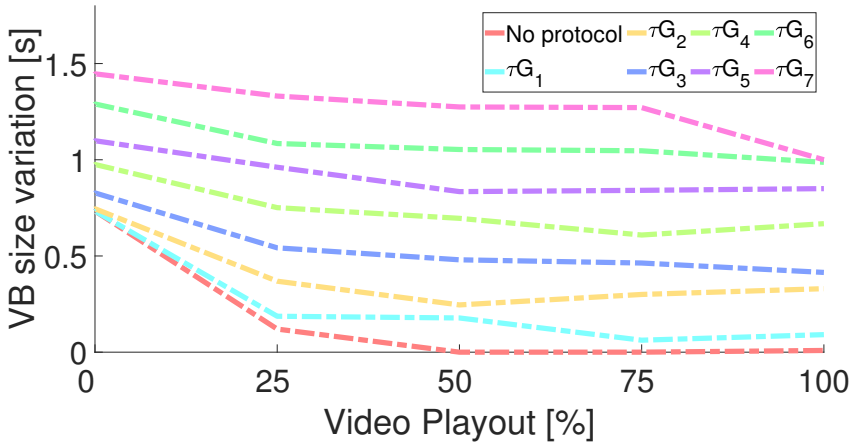


Figure 4.7: Variations in video buffer size relative to playout percentage for Scenario 02, demonstrating ASViS performance in a network environment with RTT of 0.1 seconds, packet loss rate of 0.01, and a 3-second buffer.

finally from τ_3 to τ_4. It is crucial to recognize that these intervals are sequential and do not intersect. The overall time span between the deadline and IG_4 is the collective sum of τ_2, τ_3, and τ_4. Based on the fixed conditions of the RTT and packet loss, optimal performance metrics were identified as τ_1 0 RTT, τ_2 0 RTT, τ_3 1 RTT, and τ_4 2 RTT (RTT of 0.05 s) for the layers BI, BB, EI, and EB, respectively.

For Experiment 4, a hypothesis was formulated stating that the gaps in layered-based deadlines (represented by τ values) are dependent on and proportional to the layer sizes. The video detailed in Table 4.1 features relatively small base layers and large enhancement layers due to quantization in each layer. To test this hypothesis, the configuration was reversed (i.e., small enhancement layers and large base layers) for testing purposes and used the network conditions from Experiment 3. The same three-dimensional maximum value search using the

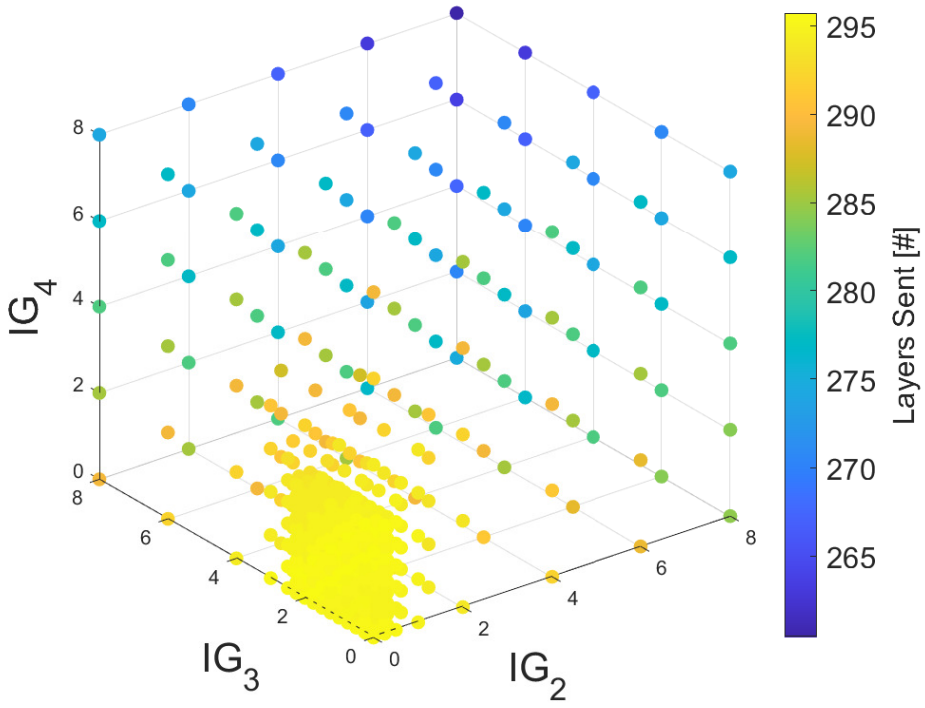


Figure 4.8: Video quality results of MM method for BB, EI, and EB layers for experiment 3.

MM method was then performed. The details of each layer size configuration (LSC) can be found in Table 4.5. This table represents the average accumulated size per layer in a GOP structure of IB×7. LSC_1 corresponds to Experiment 3, while LSC_2 is the same video from Table 4.1, but with a quantization of 20 for both BL and EL.

Table 4.5: Average layer size for both LSCs.

Layer	LSC_1[kB]	LSC_2[kB]
BI	6.97	19.31
BB	36.13	23.43
EI	0.46	1.73
EB	10.29	6.66

Performance, based on the percentage of sent layer outcomes and derived using the MM method for LSC_2, is presented in Figure 4.9. A preliminary comparison of the two 3D coordinates results, Figure 4.8 and Figure 4.9, reveals significant differences in the optimal τ values between the two LSCs. Detailed coordinates, referencing τ_2, τ_3, and τ_4, are provided in Table 4.6.

A thorough examination of the results between LSC_1 and LSC_2, and the corresponding layer sizes, has revealed a distinct association between the τ values and the sizes of layers. As depicted in Figure 4.10, the normalized layer sizes and the τ values for scenarios LSC_1 and LSC_2 are juxtaposed. It has been discerned from the blue markers that layer 2 holds a predominant role in both configurations. Conversely, the red markers have indicated that

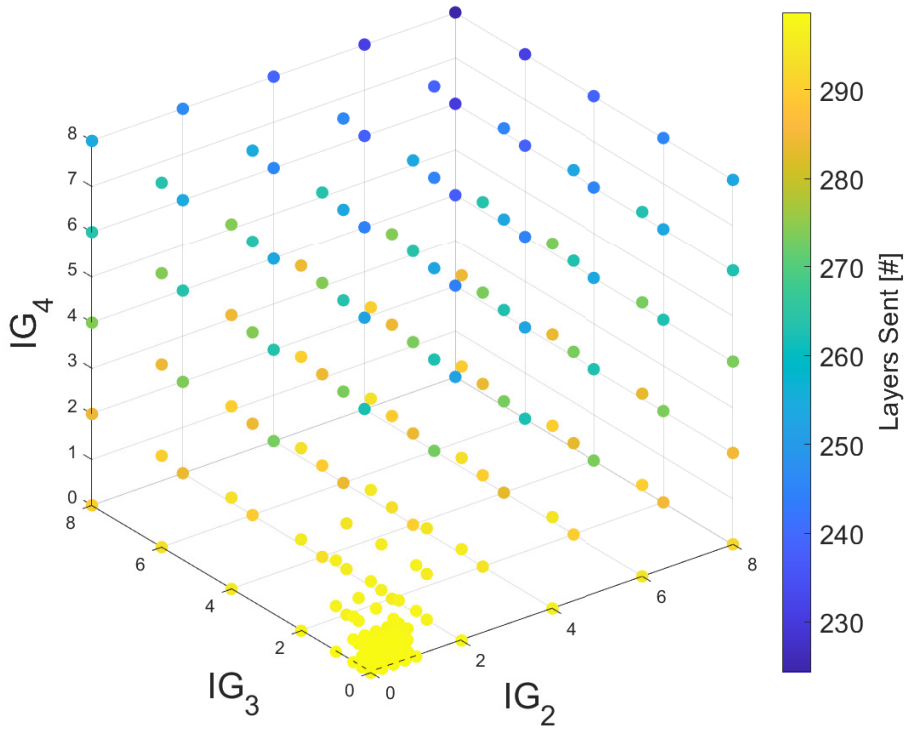


Figure 4.9: Video quality results of MM method for BB, EI, and EB layers for LSC_2 of experiment 4.

Table 4.6: Video quality results and τG coordinates for different layer size configurations.

	τG [RTT]			Sent Layers [%]			
τG	τ_2 - BB	τ_3 - EI	τ_4 - EB	BI	EI	BB	EB
LSC_1	0	1	2	100	96.6	60.2	26.6
LSC_2	0	0.5	0.75	98.7	94.53	42.7	21.4

RTT of 0.05 s.

elevated τ values are associated with more diminutive layers. A noticeable inverse correlation between the layer sizes and τ values is evident: as the magnitude of a layer amplifies, the optimal τ for video quality is observed to wane. The pivotal influence of layer sizes on video quality, in the context of τ, is underscored by this relationship.

In the fifth experiment, a comparison was made between the performance of the ABR algorithms, MPC and ASViS, using various parameters. Traditional TCP behavior, which is typical for ABR algorithms, was exhibited under the network conditions for MPC. In contrast, UDP with flow control was employed by ASViS, consistent with previous experiments. Network transmission parameters from Experiment 3 were replicated, and the τG values for the ASViS protocol were derived from the same experiment. 100 simulations were executed for both ABR algorithms, and the results presented were the mean values obtained from all

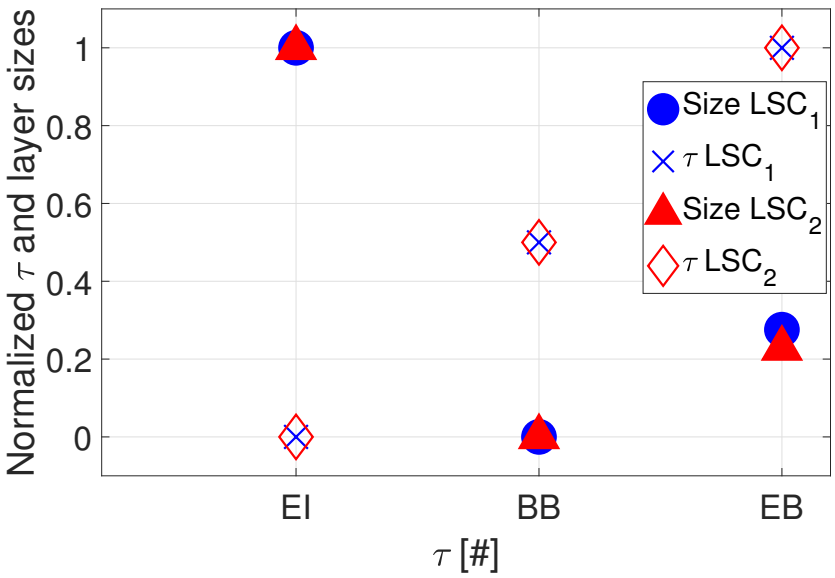


Figure 4.10: Comparison between normalized layer sizes and optimal τ ranges.

simulations.

A chunk size of 4 GOP was designated for MPC, equivalent to 32 frames or roughly 1 second of playout, given a frame rate of 30 fps. Four bitrate levels, labeled as R, were used, each integrating various layers. Comprehensive details of each R can be accessed in Table 4.7. An average across all chunks served as a reference for AVQ of Y-PSNR in this MPC implementation, accompanied by a chunk size ($d(K)$) specified in bits. The initial T_s spanned from 0.1 s to 5 s, incremented by 0.2 s. The estimated throughput (C) was calculated as the harmonic mean throughput of the previous five chunks, with weight assignments of $\lambda = 1$, $\beta = 3000$, and $\beta_s = 3000$, aligning with the recommendations in [106].

Table 4.7: Chunk properties of each bitrate level R.

	Layer				Metrics	
	BI	BB	EI	EB	AVQ Y-PSNR [dB]	$d(R)$ [Mb]
R_1	X	X	-	-	28.4	0.32
R_2	X	X	X	-	37.7	1.45
R_3	X	X	-	X	31.0	2.70
R_4	X	X	X	X	41.6	3.83

As illustrated in Figure 4.11, the relationship between the VB size variation and video playout was mapped. Results indicated that a mean VB size of 1.5 s with a standard deviation of 0.5 s was achieved by MPC, whereas ASViS achieved a mean of 0.45 s with a deviation of 0.08 s. Notably, greater stability after 20% of video playout was observed with ASViS, despite its reduced mean VB size.

Figure 4.12 portrays the anticipated bitrate distribution for both MPC and ASViS. This

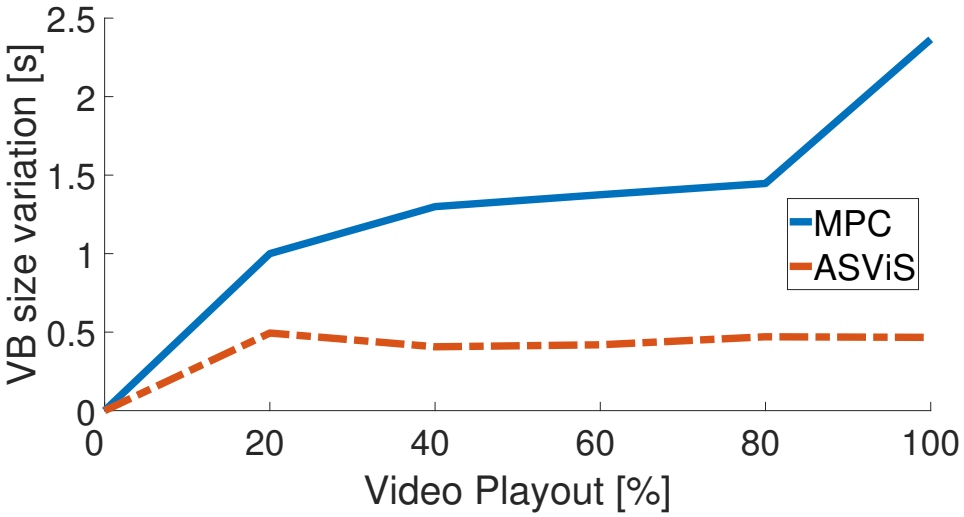


Figure 4.11: VB size comparison through video playout for ASViS and MPC.

visualization aids in comprehending the adaptability of each algorithm to network conditions. An in-depth analysis confirmed that the bitrate estimated distribution for ASViS predominantly focuses on values higher than those for MPC, a pattern that contrasts with its mean and standard deviation provided in Table 4.8.

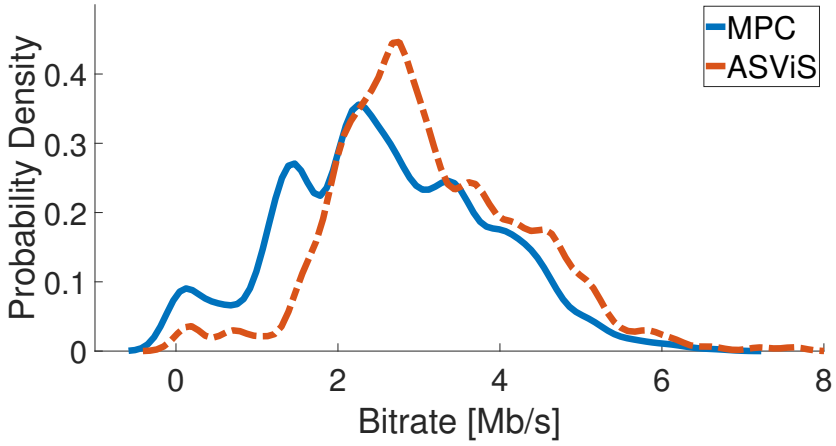


Figure 4.12: Estimated bitrate distribution for ASViS and MPC.

Subsequent evaluations explored video quality throughout the entire video playback. The Y-PSNR findings are presented in Figure 4.13, whereas VMAF results can be found in Figure 4.14. Both algorithms exhibited comparable video quality from frames 40 to 130. However, at the commencement and conclusion of the video playback, the quality of MPC was observed to be inferior to that of ASViS. This discrepancy is attributed to the conservative approach of MPC and its slower adaptation to network conditions compared to ASViS.

Furthermore, a noteworthy distinction between the results of frames 74 and 84 was observed in Figure 4.14, owing to the capability of VMAF to discern details overlooked by Y-PSNR. Detailed analysis revealed that frames 74 and 84, corresponding to a B4 frame as

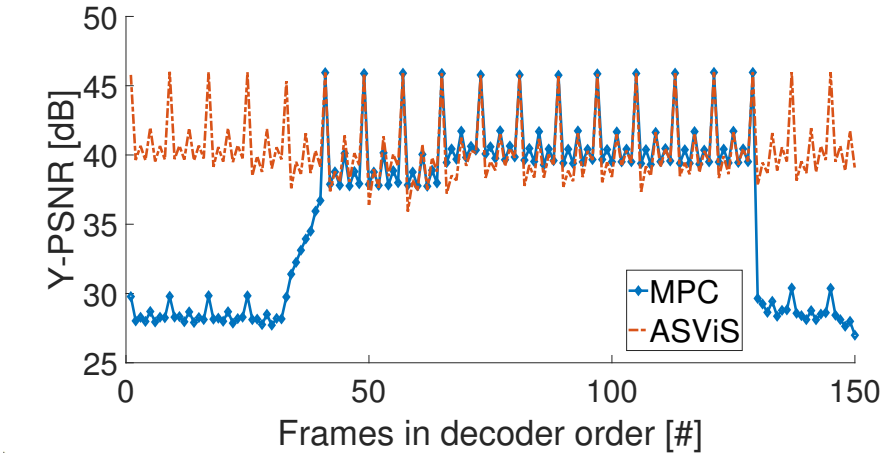


Figure 4.13: Video quality behavior of Y-PSNR for ASViS and MPC.

cross-referenced with Figure 2.4b, were dropped by ASViS.

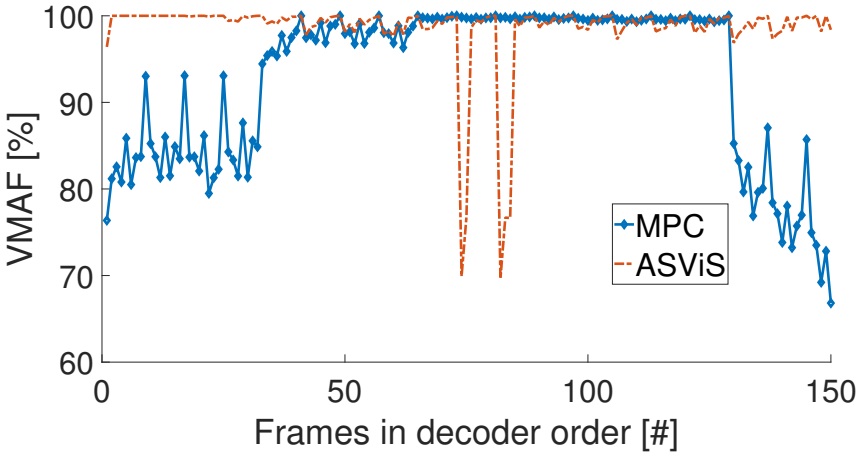


Figure 4.14: Video quality behavior of VMAF for ASViS and MPC.

The video quality metrics, presented in Figure 4.13 and Figure 4.14, were analyzed using a distinctive approach involving a boxplot and its corresponding histogram. The boxplot delineates the median, the 25th percentile at the bottom, and the 75th percentile at the top. Whiskers extend to cover values not deemed outliers. Notably, since no outliers were observed, there are no '+' markers. The results for Y-PSNR and VMAF are displayed in Figure 4.15 and Figure 4.16 respectively. Although Y-PSNR and VMAF are founded on different philosophies, both metrics consistently show the superiority of ASViS over MPC. For a comprehensive understanding of the mean and deviation values of Y-PSNR, VMAF, and bitrate, one can refer to Table 4.8. It becomes evident that, in terms of VMAF, Y-PSNR, and bitrate, the ASViS algorithm holds an edge over MPC.

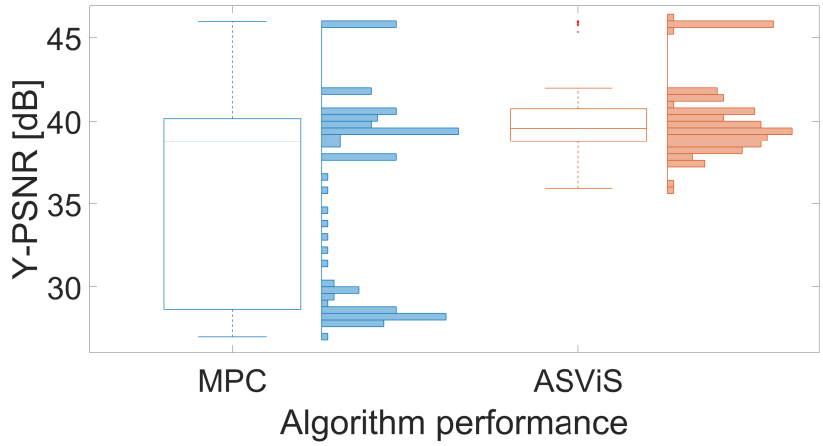


Figure 4.15: Boxplot and histogram for Y-PSNR comparison of ASViS and MPC.

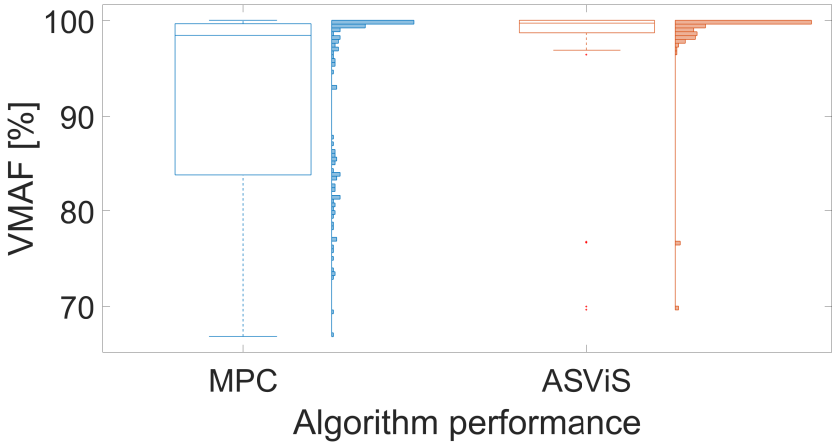


Figure 4.16: Boxplot and histogram for a VMAF comparison of ASViS and MPC.

Table 4.8: Detailed results of experiment 5.

	ASViS		MPC	
	mean	std	mean	std
Y-PSNR [dB]	41.1	2.4	39.2	6.0
VMAF [%]	98.5	4.7	92.8	9.0
Bitrate [Mb/s]	3.1	1.2	2.6	1.3

4.2. Discussion

The initial experiment demonstrates a close alignment between theoretical and observed behaviors of ASViS, with mean absolute percentage errors ranging between 6.5% and 14.2%. These findings substantiate the capability of the model to predict the experimental behavior

of ASViS, thus eliminating the need for time-consuming emulations or tests.

In the comparative assessment of Experiment 2, the ASViS protocol demonstrates proficient management of video quality under varying network conditions. Scenario 1, characterized by a lower RTT and a smaller buffer, illustrates the capacity of ASViS to ensure uninterrupted streaming while maintaining commendable quality. In contrast, Scenario 2, with a higher RTT and a larger buffer, reveals the ability of the protocol to manage buffer size effectively, using various τG configurations to prevent buffer depletion and reduce playback disruptions. Collectively, these outcomes highlight the adaptability of ASViS in enhancing the viewing experience across a spectrum of network environments without substantial compromise in video quality.

The third experiment leverages the MM method to discern specific τ coordinates that yield improved video quality under given network conditions. It reveals that optimal video quality can span a broad range of τ values and might be envisioned as a multi-dimensional solution.

In the fourth experiment, a pronounced inverse correlation is found between the τ values and layer sizes. This discovery is pivotal, suggesting that service providers can sidestep the task of pinpointing optimal values for individual videos. They can instead modify layer deadlines in proportion to layer sizes, ensuring nearly optimal outcomes.

Findings of the fifth experiment spotlight the superiority of ASViS over MPC in multiple aspects. ASViS records a 5.8% and 4.6% higher Y-PSNR and VMAF, respectively, than MPC. Moreover, ASViS displays a consistently compact video buffer size compared to MPC. It also boasts more efficient network utilization, with a bitrate surpassing that of MPC, and steadier performance, reflected in a reduced standard deviation. In conclusion, these promising results position ASViS as a skillful ABR algorithm.

ASViS outperforms traditional techniques due to its advanced data management and scalability. By employing an algorithm that prioritizes information based on layer-discarding policies and deadline-sensitive criteria, alongside the strategic use of UDP, ASViS optimizes video transmission efficiently adapting to network conditions. Its foundation on SVC ensures adaptability to future resolutions and technologies, maintaining its relevance. Experiments highlight its effectiveness, showing significant improvements in Y-PSNR and VMAF, and more efficient network utilization. This blend of adaptability, efficient flow management, and optimized layer selection policies contributes to its superiority in adaptive video streaming.

5. Conclusion

In the current landscape, robust ABR algorithms are vital to meet the vast demands of video streaming on global networks. These algorithms must ensure superior video quality, prevent stalls, and adapt swiftly to network fluctuations. The introduction of the novel concept of ASViS has been demonstrated through experiments to efficiently control video stream flow in bandwidth-limited scenarios. When SVC integrates with ASViS, it not only enhances the user QoE but also reduces network congestion. Such improvements arise from aligning the encoding rate with available throughput and discarding less critical layers.

Based on the tests conducted and the information gathered, it is suggested that the integration of DASH with ASViS enables the server to better adapt to real-time conditions, transmitting only the layers that the client can handle according to their bandwidth capacity. The ASViS proposal is found to be dynamically adaptable to network conditions and device capabilities, allowing for a more efficient use of bandwidth and an enhanced user experience, reducing start-up times and rebuffering. Furthermore, in terms of storage, it is necessary for the server to store only a single version of the scalable video. DASH is agnostic in terms of encoding, so the integration of SVC does not necessitate significant changes to the existing DASH infrastructure. Likewise, the new generation of video codec solutions, such as H.265, VP9, and AOMedia Video 1, inherently possess the scalable encoding model.

In the conclusions drawn, ASViS is portrayed as a transformative solution to ABR algorithms, with inherent limitations of DASH being addressed. Through the experiments conducted, the proficiency of ASViS in managing video stream flow, enhancing user QoE, and mitigating network congestion was validated. A noteworthy alignment was observed between theoretical and experimental behaviors, as evidenced by the mean absolute percentage errors, further emphasizing the reliability of ASViS. With its dynamic adaptability, bandwidth utilization was optimized, video quality was elevated, and interruptions were minimized, setting it apart from algorithms like MPC. In the tested scenarios, the efficacy of ASViS was demonstrated, marking a significant advancement in video streaming.

Upon reflection, the specific objectives set at the beginning of this research have been successfully met. The implementation and evaluation of the ASViS protocol demonstrated its capability to optimize adaptive scalable video transmission under fluctuating network conditions, fulfilling the first specific objective. Through the modeling of ASViS, we addressed the second objective, showcasing how the protocol effectively manages packet discards to maintain video quality. Finally, the optimization of ASViS settings, particularly the adjustment of the τ parameter, has met the third specific objective, optimizing bandwidth use and enhancing user experience in variable network scenarios. These achievements underscore the success of ASViS in tackling the challenges posed by video transmission in the current internet era.

In the conclusion of our study, we underscore the significant technical achievements of ASViS, highlighting its innovative approach to adaptive scalable video streaming. Key accomplishments include the establishment of a cross-layer solution that enhances video quality under fluctuating network conditions, a strategic implementation of layer-discarding policies that optimizes bandwidth utilization, and the successful adaptation to future video resolutions and technologies through its foundational reliance on SVC. These advancements

not only demonstrate the superior performance of ASViS over traditional ABR streaming methods but also its readiness to meet the evolving demands of digital video distribution. Collectively, these technical achievements underscore the potential of ASViS to revolutionize video streaming by offering a more efficient, adaptable, and quality-focused solution.

5.1. Future Work

In specific experimental contexts, the potential of ASViS has been demonstrated, yet its performance across broader and more challenging scenarios is anticipated to be an area of significant interest. In the realm of 5G mobile networks, characterized by high download speeds and reduced latency, it is hypothesized that ultra-high-definition video streaming with minimal interruptions could be facilitated by ASViS. In contrast, challenges are expected in regions with underdeveloped internet infrastructure or in satellite networks with inherent high latency. In such scenarios, it is theorized that smooth, albeit lower-quality video playback, might be prioritized by ASViS. Further exploration is deemed necessary to ascertain the resilience of ASViS in conditions that are less than ideal.

A crucial direction for future research involves the comparative analysis of ASViS with other leading ABR protocols, such as FESTIVE and Pensieve. These protocols represent significant benchmarks in the field of adaptive bitrate streaming. Through comprehensive comparative studies, the performance of ASViS against these protocols is to be assessed, particularly focusing on efficiency, adaptability, and quality of service under similar network conditions. Such comparisons are considered vital to identify areas where ASViS either excels or requires improvements, offering insights crucial for its further development.

Furthermore, the conduct of more exhaustive tests on ASViS is proposed, involving videos with varying resolutions and different frame rates. These tests are to be designed to emulate realistic network conditions, particularly those prevalent in mobile networks. Insights gained from such extensive testing will be instrumental in refining ASViS, ensuring its robustness and effectiveness as a solution in the evolving landscape of video streaming technologies.

Bibliography

[1] E. Coronado, J. M. V. Millán, and A. Garrido, "Improvements to multimedia content delivery over ieee 802.11 networks," *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, pp. 1–6, 2020.

[2] L. Eggert, G. Fairhurst, and G. Shepherd, "Udp usage guidelines," BCP 145, RFC Editor, March 2017. Available online: https://www.rfc-editor.org/rfc/rfc8085.html (accessed on 01 november 2023).

[3] Comcast, "Covid-19 network update," Dec 2020. Available online: https://corporate.comcast.com/covid-19/network/may-20-2020 (accessed on 01 november 2023).

[4] A. Mercat, A. Mäkinen, J. Sainio, A. Lemmetti, M. Viitanen, and J. Vanne, "Comparative rate-distortion-complexity analysis of vvc and hevc video codecs," *IEEE Access*, vol. 9, pp. 67813–67828, 2021.

[5] T. Tang, A. Abuhmaid, M. Olaimat, D. M. Oudat, M. Aldhaeebi, and E. Bamanger, "Efficiency of flipped classroom with online-based teaching under covid-19," *Interactive Learning Environments*, vol. 31, no. 2, pp. 1077–1088, 2023.

[6] Y.-H. Hu, "Effects of the covid-19 pandemic on the online learning behaviors of university students in taiwan," *Education and Information Technologies*, vol. 27, p. 469–491, 09 2021.

[7] A. O. El Meligy, M. S. Hassan, and T. Landolsi, "A buffer-based rate adaptation approach for video streaming over http," in *2020 Wireless Telecommunications Symposium (WTS)*, pp. 1–5, 2020.

[8] Sandvine, "Sandvine's 2023 global internet phenomena report," June 2023. Available online: https://www.sandvine.com/hubfs/Sandvine_Redesign_2019/Downloads/2023/reports/Sandvine%20GIPR%202023.pdf (accessed on 01 november 2023).

[9] Ericsson, "Ericsson mobility report," June 2023. Available online: https://www.ericsson.com/49dd9d/assets/local/reports-papers/mobility-report/documents/2023/ericsson-mobility-report-june-2023.pdf (accessed on 01 november 2023).

[10] J. Pearce, "Netflix and youtube to reduce stream quality in europe due to coronavirus," Mar 2020. Available online: https://www.ibc.org/publish/netflix-and-youtube-to-reduce-stream-quality-in-europe-due-to-coronavirus/5615.article (accessed on 01 november 2023).

[11] J. Zhao, J. Liu, C. Zhang, Y. Cui, Y. Jiang, and W. Gong, "Mptcp+: Enhancing adaptive http video streaming over multipath," in *2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*, pp. 1–6, 2020.

[12] J. Duan, M. Zhang, J. Wang, S. Han, X. Chen, and X. Yang, "Vcc-dash: A video content complexity-aware dash bitrate adaptation strategy," *Electronics*, vol. 9, no. 2, 2020.

[13] S. M. A. H. Bukhari, W. Afandi, M. U. S. Khan, T. Maqsood, M. B. Qureshi, M. A. B.

Fayyaz, and R. Nawaz, "E-ensemble: A novel ensemble classifier for encrypted video identification," *Electronics*, vol. 11, no. 24, 2022.

[14] T. Matsumoto and M. Yamamoto, "Fairness improvement by combination of abr and tcp algorithms in abr video streaming," *IEICE Communications Express*, vol. 10, no. 5, pp. 225–230, 2021.

[15] A. Hodroj, M. Ibrahim, and Y. Hadjadj-Aoul, "A survey on video streaming in multi-path and multihomed overlay networks," *IEEE Access*, vol. 9, pp. 66816–66828, 2021.

[16] Q. Yanyuan, S. Hao, K. Pattipati, F. Qian, S. Sen, and C. Yue, "Abr streaming of vbr-encoded videos: characterization, challenges, and solutions," pp. 366–378, 12 2018.

[17] C. Kreuzberger, D. Posch, and H. Hellwagner, "A scalable video coding dataset and toolchain for dynamic adaptive streaming over http," in *Proceedings of the 6th ACM Multimedia Systems Conference*, MMSys '15, (New York, NY, USA), p. 213–218, Association for Computing Machinery, 2015.

[18] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer, and R. Zimmermann, "A survey on bitrate adaptation schemes for streaming media over http," *IEEE Communications Surveys Tutorials*, vol. 21, no. 1, pp. 562–585, 2019.

[19] J. Kua, G. Armitage, P. Branch, and J. But, "Adaptive chunklets and aqm for higher-performance content streaming," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 15, Dec. 2019.

[20] Z. Meng, J. Chen, Y. Guo, C. Sun, H. Hu, and M. Xu, "Pitree: Practical implementation of abr algorithms using decision trees," pp. 2431–2439, 10 2019.

[21] K. Khan and W. Goodridge, "Ultra-hd video streaming in 5g fixed wireless access bottlenecks," in *Proceedings of CECNet 2021 - The 11th International Conference on Electronics, Communications and Networks (CECNet), November 18-21, 2021, Virtual Event / Beijing, China* (A. J. Tallón-Ballesteros, ed.), vol. 345 of *Frontiers in Artificial Intelligence and Applications*, pp. 515–524, IOS Press, 2021.

[22] K. Ragimova, V. Loginov, and E. Khorov, "Analysis of youtube dash traffic," in *2019 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, pp. 1–5, 2019.

[23] W. Choi and J. Yoon, "Sate: Providing stable and agile adaptation in http-based video streaming," *IEEE Access*, vol. 7, pp. 26830–26841, 2019.

[24] J. Clement, "Devices used to watch online video worldwide 2019," Nov 2020. Available online: https://www.statista.com/statistics/784351/online-video-devices/ (accessed on 01 november 2023).

[25] Y. Qin, S. Sen, and B. Wang, "Abr streaming with separate audio and video tracks: Measurements and best practices," in *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, CoNEXT '19, (New York, NY, USA), p. 158–164, Association for Computing Machinery, 2019.

[26] M. Iwamoto, T. Otoshi, D. Kominami, and M. Murata, "Rate adaptation with bayesian attractor model for mpeg-dash," in *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 0859–0865, 2019.

[27] T. Huang, C. Zhou, R.-X. Zhang, C. Wu, X. Yao, and L. Sun, "Stick: A harmonious fusion of buffer-based and learning-based approach for adaptive streaming," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pp. 1967–1976, 2020.

[28] Y. Qin, S. Hao, K. R. Pattipati, F. Qian, S. Sen, B. Wang, and C. Yue, "Quality-aware strategies for optimizing abr video streaming qoe and reducing data usage," in *Proceedings of the 10th ACM Multimedia Systems Conference*, MMSys '19, (New York, NY, USA), p. 189–200, Association for Computing Machinery, 2019.

[29] J.-M. Martinez-Caro and M.-D. Cano, "On the identification and prediction of stalling events to improve qoe in video streaming," *Electronics*, vol. 10, no. 6, 2021.

[30] T. Zhang and S. Mao, "An overview of emerging video coding standards," *GetMobile: Mobile Comp. and Comm.*, vol. 22, p. 13–20, may 2019.

[31] S. Subbarayappa and K. R. Rao, "Video quality evaluation and testing verification of h.264, hevc, vvc and evc video compression standards," *IOP Conference Series: Materials Science and Engineering*, vol. 1045, p. 012028, feb 2021.

[32] A. Sanhueza, H. Méric, and C. Estevez, "Efficient video streaming rate control based on a deadline-sensitive selection of svc layers," in *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 502–506, 2017.

[33] M. Nguyen, H. Amirpour, C. Timmerer, and H. Hellwagner, "Scalable high efficiency video coding based http adaptive streaming over quic," in *Proceedings of the Workshop on the Evolution, Performance, and Interoperability of QUIC*, EPIQ '20, (New York, NY, USA), p. 28–34, Association for Computing Machinery, 2020.

[34] A. Elgabli, K. Liu, and V. Aggarwal, "Optimized preference-aware multi-path video streaming with scalable video coding," *IEEE Transactions on Mobile Computing*, vol. 19, no. 1, pp. 159–172, 2020.

[35] D. Wang, Y. Sun, C. Zhu, W. Li, F. Dufaux, and J. Luo, "Fast depth and mode decision in intra prediction for quality shvc," *IEEE Transactions on Image Processing*, vol. 29, pp. 6136–6150, 2020.

[36] X. HoangVan, "Improving shvc performance with a block based joint layer prediction solution," in *2018 International Workshop on Advanced Image Technology (IWAIT)*, pp. 1–4, 2018.

[37] L. Balaji, A. Dhanalakshmi, and C. Raja, "Scalable extension of hevc with adaptive deblocking filter for bandwidth-limited applications," *ICT Express*, vol. 8, no. 3, pp. 388–395, 2022.

[38] F. Zhang, A. V. Katsenou, M. Afonso, G. Dimitrov, and D. Bull, "Comparing vvc, hevc and av1 using objective and subjective assessments," *arXiv: Image and Video Processing*, 2020.

[39] D. Lorenzi, "Qoe- and energy-aware content consumption for http adaptive streaming," in *Proceedings of the 14th Conference on ACM Multimedia Systems*, MMSys '23, (New York, NY, USA), p. 348–352, Association for Computing Machinery, 2023.

[40] Y. Chen, D. Mukherjee, J. Han, A. Grange, Y. Xu, S. Parker, C. Chen, H. Su, U. Joshi,

C.-H. Chiang, and et al., "An overview of coding tools in av1: the first video codec from the alliance for open media," *APSIPA Transactions on Signal and Information Processing*, vol. 9, p. e6, 2020.

[41] P. Rivaz, J. Haughton, A. Grange, and L. Quillio, *AV1 Bitstream and Decoding Process Specification*. The Alliance for Open Media, 08 2019. Available online: https://aomediacodec.github.io/av1-spec/ (accessed on 01 november 2023).

[42] A. Pennington, "Codec wars: The battle between hevc and av1," Mar 2018. Available online: https://www.ibc.org/trends/codec-wars-the-battle-between-hevc-and-av1/2710.article (accessed on 01 november 2023).

[43] J. Ozer, "A video codec licensing update," Jan 2019. Available online: https://www.streamingmedia.com/Articles/ReadArticle.aspx?ArticleID=129386 (accessed on 01 november 2023).

[44] C. Jia, F. Ye, F. Dong, K. Lin, L. Chiariglione, S. Ma, H. Sun, and W. Gao, "Mpai-eev: Standardization efforts of artificial intelligence based end-to-end video coding," *ArXiv*, vol. abs/2309.07589, 2023.

[45] L. Tang, X. Zhang, G. Zhang, and X. Ma, "Scene matters: Model-based deep video compression," *ArXiv*, vol. abs/2303.04557, 2023.

[46] G.-H. Wang, J. Li, B. Li, and Y. Lu, "EVC: Towards real-time neural image compression with mask decay," in *The Eleventh International Conference on Learning Representations*, vol. abs/2302.05071, 2023.

[47] Y. Sani, D. Raca, J. J. Quinlan, and C. J. Sreenan, "Smash: A supervised machine learning approach to adaptive video streaming over http," in *2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX)*, pp. 1–6, 2020.

[48] M. Behzadpour and M. Ghanbari, "Improving precision of objective image/video quality meters," *Multimedia Tools and Applications*, vol. 82, pp. 1–14, 07 2022.

[49] J. Du, "Aivmaf: Automatic image quality estimation based on improved vmaf and yolov4," *Journal of Physics: Conference Series*, vol. 2289, p. 012020, jun 2022.

[50] S. Heine, F. Völk, R. T. Schwarz, and A. Knopp, "Concept and evaluation of 5g backhauling via starlink," in *39th International Communications Satellite Systems Conference (ICSSC 2022)*, vol. 2022, pp. 69–75, 2022.

[51] J. Garcia, S. Sundberg, G. Caso, and A. Brunstrom, "Multi-timescale evaluation of starlink throughput," in *Proceedings of the 1st ACM Workshop on LEO Networking and Communication*, LEO-NET '23, (New York, NY, USA), p. 31–36, Association for Computing Machinery, 2023.

[52] K. Kaur, M. Kaur, K. Kaur, and A. Madaan, "A comparative study of osi and tcp/ip models," *International Journal of Engineering and Management Research*, vol. 13, p. 127–135, Apr. 2023.

[53] M. Conti, A. Gangwal, M. Hassan, C. Lal, and E. Losiouk, "The road ahead for networking: A survey on icn-ip coexistence solutions," *IEEE Communications Surveys Tutorials*, vol. 22, no. 3, pp. 2104–2129, 2020.

[54] C. Onyia, K. Nnamani, E. Alagbu, and C. Ezeagwu, "Comparative analysis of osi and

tcp/ip models in network communication," *Quest*, vol. 7, pp. 08–14, 07 2021.

[55] G. Olmedo, R. Lara-Cueva, D. Martínez, and C. Almeida, "Performance analysis of a novel tcp protocol algorithm adapted to wireless networks," *Future Internet*, vol. 12, p. 101, 06 2020.

[56] M. J. Khan, S. Harous, and A. Bentaleb, "Client-driven adaptive bitrate techniques for media streaming over http: Initial findings," in *2020 IEEE International Conference on Electro Information Technology (EIT)*, pp. 053–059, 2020.

[57] T. Chowdhury and M. Alam, "Performance evaluation of tcp vegas over tcp reno and tcp newreno over tcp reno," *JOIV : International Journal on Informatics Visualization*, vol. 3, 08 2019.

[58] G. Mahfoudi, F. Retraint, F. Morain-Nicolier, and M. M. Pic, "Statistical h.264 double compression detection method based on dct coefficients," *IEEE Access*, vol. 10, pp. 4271–4283, 2022.

[59] K. Nihei, H. Yoshida, N. Kai, K. Satoda, and K. Chono, "Adaptive bitrate control of scalable video for live video streaming on best-effort network," in *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7, 2018.

[60] W. Penny, D. Palomino, M. Porto, B. Zatt, and L. Indrusiak, "Performance evaluation of hevc rcl applications mapped onto noc-based embedded platforms," in *2019 32nd Symposium on Integrated Circuits and Systems Design (SBCCI)*, pp. 1–6, 2019.

[61] H. Zhao, Y. Liu, Y. Wang, S. Liu, and C. Feng, "A video steganography method based on transform block decision for h.265/hevc," *IEEE Access*, vol. 9, pp. 55506–55521, 2021.

[62] C.-T. Ni, Y.-C. Huang, and P.-Y. Chen, "A hardware-friendlyand high-efficiency h.265/hevc encoder for visual sensor networks," *Sensors*, vol. 23, no. 5, 2023.

[63] X. Lu, C. Yu, and X. Jin, "A fast hevc intra-coding algorithm based on texture homogeneity and spatio-temporal correlation," *EURASIP Journal on Advances in Signal Processing*, vol. 2018, 12 2018.

[64] C. Ma, D. Liu, X. Peng, and F. Wu, "Convolutional neural network-based arithmetic coding of dc coefficients for hevc intra coding," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, pp. 1772–1776, 2018.

[65] R. Skupin, C. Bartnik, A. Wieckowski, Y. Sanchez, B. Bross, C. Hellge, and T. Schierl, "Open gop resolution switching in http adaptive streaming with vvc," in *2021 Picture Coding Symposium (PCS)*, pp. 1–5, 2021.

[66] R. Pourreza and T. Cohen, "Extending neural p-frame codecs for b-frame coding," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, (Los Alamitos, CA, USA), pp. 6660–6669, IEEE Computer Society, oct 2021.

[67] Y. Wu, X. Bai, Y. Hu, and M. Chen, "A novel video transmission latency measurement method for intelligent cloud computing," *Applied Sciences*, vol. 12, no. 24, 2022.

[68] P. Topiwala, W. Dai, J. Pian, K. Biondi, and A. Krovvidi, "Video quality analysis: Steps towards unifying full and no reference cases," *Standards*, vol. 2, no. 3, pp. 402–416, 2022.

[69] A. Antsiferova, A. Yakovenko, N. Safonov, D. Kulikov, A. Gushchin, and D. Vatolin, "Applying objective quality metrics to video-codec comparisons: Choosing the best metric for subjective quality estimation," pp. 199–210, 11 2021.

[70] F. Zhang, K. Kurokawa, A. Lassoued, J. Crowell, and D. Miller, "Cone photoreceptor classification in the living human eye from photostimulation-induced phase dynamics," *Proceedings of the National Academy of Sciences*, vol. 116, p. 201816360, 04 2019.

[71] E. Lee, L. Hsu, E. Chen, and C. Lee, "Cross-resolution flow propagation for foveated video super-resolution," in *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, (Los Alamitos, CA, USA), pp. 1766–1775, IEEE Computer Society, jan 2023.

[72] J. Vlaović, S. Rimac-Drlje, F. Vranješ, and R. P. Kovač, "Evaluation of adaptive bitrate selection algorithms for mpeg dash," in *2019 International Symposium ELMAR*, pp. 73–76, 2019.

[73] N. T. Blog, "Per-title encode optimization," Apr 2017. Available online: https://netflixtechblog.com/per-title-encode-optimization-7e99442b62a2 (accessed on 01 november 2023).

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

[75] O. Keleş, M. A. Yılmaz, A. M. Tekalp, C. Korkmaz, and Z. Dogan, "On the computation of psnr for a set of images or video," 2021.

[76] A. Thakur, "SORT: A System for Adaptive Transmission of Video Over Delay Tolerant Networks," *International Journal of Wireless Networks and Broadband Technologies (IJWNBT)*, vol. 9, pp. 22–49, July 2020.

[77] N. T. Blog, "Vmaf: The journey continues," Oct 2018. Available online: https://netflixtechblog.com/vmaf-the-journey-continues-44b51ee9ed12 (accessed on 01 november 2023).

[78] C. G. Bampis, Z. Li, I. Katsavounidis, and A. C. Bovik, "Recurrent and dynamic models for predicting streaming video quality of experience," *IEEE Transactions on Image Processing*, vol. 27, no. 7, pp. 3316–3331, 2018.

[79] Z. Luo, C. Zhu, Y. Huang, R. Xie, L. Song, and C.-C. J. Kuo, "Vmaf oriented perceptual coding based on piecewise metric coupling," *IEEE Transactions on Image Processing*, vol. 30, pp. 5109–5121, 2021.

[80] L.-H. Chen, C. G. Bampis, Z. Li, J. Sole, and A. C. Bovik, "Perceptual video quality prediction emphasizing chroma distortions," *IEEE Transactions on Image Processing*, vol. 30, pp. 1408–1422, 2021.

[81] R. Behravesh, A. Rao, D. F. Perez-Ramirez, D. Harutyunyan, R. Riggio, and M. Boman, "Machine learning at the mobile edge: The case of dynamic adaptive streaming over http (dash)," *IEEE Transactions on Network and Service Management*, vol. 19, no. 4, pp. 4779–4793, 2022.

[82] J. Uriol, I. Yeregui, Gabilondo, R. Viola, P. Angueira, and J. Montalbán, "Context-

aware adaptive prefetching for dash streaming over 5g networks," in *2023 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, pp. 1–6, 2023.

[83] N. Souane, M. Bourenane, and Y. Douga, "Deep reinforcement learning-based approach for video streaming: Dynamic adaptive video streaming over http," *Applied Sciences*, vol. 13, no. 21, 2023.

[84] R. Ul Mustafa, D. Moura, and C. E. Rothenberg, "Machine learning approach to estimate video qoe of encrypted dash traffic in 5g networks," in *2021 IEEE Statistical Signal Processing Workshop (SSP)*, pp. 586–589, 2021.

[85] C. Ge and N. Wang, "Real-time qoe estimation of dash-based mobile video applications through edge computing," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 766–771, 2018.

[86] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hoßfeld, and P. Tran-Gia, "A survey on quality of experience of http adaptive streaming," *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, pp. 469–492, 2015.

[87] S. Chang, K. Wang, and J. Ho, "Optimal dash video scheduling over variable-bit-rate networks," in *2018 9th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP)*, pp. 41–48, 2018.

[88] J. Vlaović, S. Rimac-Drlje, F. Vranješ, and R. P. Kovač, "Evaluation of adaptive bitrate selection algorithms for mpeg dash," in *2019 International Symposium ELMAR*, pp. 73–76, 2019.

[89] K. Spiteri, R. Sitaraman, and D. Sparacio, "From theory to practice: Improving bitrate adaptation in the dash reference player," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 15, jul 2019.

[90] S. Schwarzmann, P. Breitbach, T. Zinner, and M. Rost, "Modeling adaptive video streaming using discrete-time analysis," in *2019 31st International Teletraffic Congress (ITC 31)*, pp. 121–129, 2019.

[91] I. Sarkar, G. Urvoy-Keller, D. Lopez Pacheco, S. Roubia, and Q. Jacquemart, "Multi-Source HTTP Live Streaming (MSHLS), an ABR Algorithm for Hybrid V2V-CDN Video Streaming," research report, I3S, Université Côte d'Azur ; EasyBroadcast, Nov. 2021.

[92] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over http," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, SIGCOMM '15, (New York, NY, USA), p. 325–338, Association for Computing Machinery, 2015.

[93] K. Khan and W. Goodridge, "Server-based and network-assisted solutions for adaptive video streaming," *International Journal of Advanced Manufacturing Technology*, vol. 9, pp. 3432–3442, 12 2017.

[94] J. W. Kleinrouweler, S. Cabrero, and P. Cesar, "Delivering stable high-quality video: An sdn architecture with dash assisting network elements," in *Proceedings of the 7th International Conference on Multimedia Systems*, MMSys '16, (New York, NY, USA), Association for Computing Machinery, 2016.

[95] M. Palmer, T. Krüger, B. Chandrasekaran, and A. Feldmann, "The quic fix for optimal video streaming," *Proceedings of the Workshop on the Evolution, Performance, and Interoperability of QUIC - EPIQ'18*, 2018.

[96] A. Erfanian, F. Tashtarian, A. Zabrovskiy, C. Timmerer, and H. Hellwagner, "Oscar: On optimizing resource utilization in live video streaming," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 552–569, 2021.

[97] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," in *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, (New York, NY, USA), p. 187–198, Association for Computing Machinery, 2014.

[98] P. Juluri, V. Tamarapalli, and D. Medhi, "Sara: Segment aware rate adaptation algorithm for dynamic adaptive streaming over http," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, pp. 1765–1770, 2015.

[99] J. Vlaović, S. Rimac-Drlje, and D. Žagar, "Content dependent representation selection model for systems based on mpeg dash," *Electronics*, vol. 10, no. 15, 2021.

[100] M. Nguyen, C. Timmerer, and H. Hellwagner, "H2br: An http/2-based retransmission technique to improve the qoe of adaptive video streaming," in *Proceedings of the 25th ACM Workshop on Packet Video*, PV '20, (New York, NY, USA), p. 1–7, Association for Computing Machinery, 2020.

[101] H.-Z. Dong, Z.-C. Guo, Y.-Q. Sheng, and X.-Y. Zhu, "Pasa: Towards fair rate adaptation for http-based video streaming through server assistance," *Journal of Computers*, vol. 31, no. 3, pp. 27–39, 2020.

[102] M. Grüner, M. Licciardello, and A. Singla, "Reconstructing proprietary video streaming algorithms," in *Proceedings of the 2020 USENIX Conference on Usenix Annual Technical Conference*, USENIX ATC'20, (USA), USENIX Association, 2020.

[103] M. Licciardello, M. Grüner, and A. Singla, *Understanding Video Streaming Algorithms in the Wild*, pp. 298–313. Cham: Springer International Publishing, 03 2020.

[104] S. K. Mehr, P. Juluri, M. Maddumala, and D. Medhi, "An adaptation aware hybrid client-cache approach for video delivery with dynamic adaptive streaming over http," in *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, pp. 1–5, 2018.

[105] A. Bentaleb, P. K. Yadav, W. T. Ooi, and R. Zimmermann, "Dq-dash: A queuing theory approach to distributed adaptive video streaming," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 16, mar 2020.

[106] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over http," *SIGCOMM Comput. Commun. Rev.*, vol. 45, p. 325–338, Aug. 2015.

[107] C. Qiao, J. Wang, and Y. Liu, "Beyond qoe: Diversity adaptation in video streaming at the edge," *IEEE/ACM Transactions on Networking*, vol. 29, no. 1, pp. 289–302, 2021.

[108] Z. Deng, Y. Liu, J. Liu, and A. Argyriou, "Cross-layer dash-based multipath video streaming over lte and 802.11ac networks," vol. 80, p. 16007–16026, apr 2021.

[109] A. Yaqoob, T. Bi, and G.-M. Muntean, "A survey on adaptive 360° video streaming: Solutions, challenges and opportunities," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2801–2838, 2020.

[110] A. Sanhueza, H. Méric, and C. Estevez, "Efficient video streaming rate control based on a deadline-sensitive selection of svc layers," *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 502–506, 2017.

[111] C. Mineo, D. Lines, and D. Cerniglia, "Generalised bisection method for optimum ultrasonic ray tracing and focusing in multi-layered structures," *Ultrasonics*, vol. 111, p. 106330, 2021.

[112] R. Etesami, M. Madadi, M. Mashinchi, and R. Ganjoei, "A new method for rooting nonlinear equations based on the bisection method," *MethodsX*, vol. 8, p. 101502, 08 2021.