Chapter 2: Literature Review

2.1 Overview of Video Streaming Technology

2.1.1 Introduction to Video Streaming

Video streaming is the process of transmitting video content over a network, allowing users to watch it in real-time without downloading the entire file beforehand. This involves dividing the video into small segments, which are then transmitted sequentially and played back by the client (Sodagar, 2011). A key performance metric is latency. In the context of this research, latency encompasses not only the delay between content creation and initial playback (startup delay) and interruptions during playback (stalls/rebuffering), but also the relative differences in playback progression between concurrent viewers experiencing heterogeneous network conditions. Minimizing these forms of latency is crucial for a positive user experience, especially in today's digital world where high-quality video streaming is increasingly demanded, particularly on mobile devices. The ever-increasing demand for high-quality video streaming necessitates innovative solutions to minimize latency, defined as the delay between content creation and playback. High latency significantly impacts user engagement and potential revenue. Factors contributing to latency include network conditions (bandwidth, Round-Trip Time (RTT) / delay, packet loss), server performance, video encoding schemes, and client-side factors (buffer management, adaptation strategies). Adaptive streaming techniques are crucial for mitigating the impact of network variability (Oyman & Singh, 2012).

2.1.2 Historical Evolution of Streaming Protocols

Early video streaming solutions often struggled with maintaining consistent quality under varying network conditions. The rise of adaptive bitrate streaming addressed this challenge. Three prominent adaptive HTTP streaming protocols have emerged:

1. **MPEG-DASH (Dynamic Adaptive Streaming over HTTP):** An international standard developed by ISO/IEC, DASH provides a framework for adaptive HTTP streaming, enabling clients to dynamically adjust the bitrate based on network conditions and device capabilities (Sodagar, 2011). It utilizes segmented video delivery and allows clients to select video segments in various qualities. DASH’s focus on adaptation to diverse network conditions directly contributes to its ability to minimize latency issues.
2. **HTTP Live Streaming (HLS):** Developed by Apple, HLS is another adaptive streaming protocol that allows for dynamic adjustment of bitrate through segmented delivery over HTTP, providing video segments in multiple qualities (Li et al., 2014). The fragmented delivery inherent in HLS helps reduce the impact of transmission delays.

3. **Smooth Streaming:** Developed by Microsoft, Smooth Streaming is a segmented HTTP streaming technique providing similar capabilities to DASH and HLS for adaptive bitrate control and thus contributing to improved latency (Akhshabi et al., 2013).

These protocols differ in how they segment the video (DASH supports both fixed and variable-length segments, while HLS uses fixed-length segments), the way they signal quality levels to the client, and the methods used for adaptive bitrate selection. However, all aim to minimize latency via efficient adaptation to dynamically changing network conditions (Akhshabi et al., 2013; Li et al., 2014).

2.1.3 Adaptive Streaming Mechanisms

Adaptive HTTP streaming (like DASH and HLS) significantly improves video playback quality by dynamically adapting to fluctuating network conditions and device capabilities (Sodagar, 2011). Key mechanisms include:

1. **Adaptive Bitrate Selection:** The client continuously monitors network conditions (e.g., bandwidth, **Round-Trip Time (RTT) / latency**, packet loss) and selects a bitrate that maximizes QoE without causing buffer underruns. Algorithms for bitrate selection must consider the tradeoff between achieving high quality and preventing interruptions by dynamically balancing bandwidth and latency. This usually requires forecasting future network conditions to anticipate network limitations (Balachandran et al., 2013).
2. **Segment-Level Adaptation:** Adaptive streaming uses segmented video delivery, allowing the client to request specific video segments at different qualities based on available bandwidth and latency (Sodagar, 2011). The small segment sizes contribute to better adaptation to varying network bandwidth, which helps minimize delays.
3. **Buffer Management:** Clients use buffers to store received segments, smoothing out short-term fluctuations in network bandwidth and preventing interruptions in playback. Buffer management algorithms must consider the tradeoff between minimizing buffering delays and managing memory usage on client devices (Miller et al., 2012).

2.1.4 Role of Standards and Frameworks

Standardization has played a key role in ensuring interoperability and efficient implementation of adaptive streaming. The most significant is the MPEG-DASH standard, which provides a common framework for adaptive HTTP streaming, addressing challenges related to latency, network variability, and device heterogeneity. Other relevant standards and frameworks (e.g., HLS and Smooth Streaming) support similar adaptive streaming functionalities and contribute to the overall efficiency and reliability of video streaming over various network environments. Industry initiatives and working groups are focused on ongoing refinements of standards and protocols to improve performance and enhance user experience in mobile environments where the bandwidth can fluctuate (Oyman & Singh, 2012). Research efforts focus on innovating quality adaptation algorithms that prioritize low latency and minimize the negative impact of buffering and low resolution (Yin et al., 2015).

2.2 Factors Contributing to Latency in Video Streaming

2.2.1 Network Conditions

Network conditions significantly impact video streaming latency. Key factors include:

1. **Bandwidth:** Insufficient bandwidth leads to increased buffering and reduced video quality. Lower bandwidth necessitates lower bitrates, potentially impacting the viewing experience (Dobrian et al., 2011). Fluctuations in bandwidth, especially prevalent in mobile networks, cause significant challenges in maintaining smooth playback.
2. **Delay (Round-Trip Time - RTT):** Network delay, fundamentally characterized by the Round-Trip Time (RTT) for data packets, directly adds to overall latency. High RTT significantly impacts the agility of ABR decision loops, as feedback on throughput and buffer status is delayed. Each segment request and initial data reception is penalized by the RTT, prolonging download times even if instantaneous bandwidth is high. This is particularly critical during startup and recovery from stalls. High delay in mobile networks, particularly in broadband networks, exacerbates latency issues (Dahlman et al., 2014).
3. **Packet Loss:** Packet loss, where data packets are lost during transmission, necessitates retransmission, significantly increasing latency and potentially leading to inaccurate throughput estimations by the ABR if not properly accounted for. Mobile networks are particularly susceptible to packet loss due to signal interference and mobility (Oyman & Singh, 2012).
4. **Jitter:** Jitter, variations in the time between packet arrivals (or inter-packet delay variance), causes uneven playback and disrupts the viewing experience. High jitter can make consistent throughput prediction difficult for ABR algorithms and can lead to buffer mismanagement if not smoothed out effectively by the playout buffer. Jitter can be particularly problematic in wireless environments due to interference and signal fluctuations (Akhshabi et al., 2013).

Adaptive streaming protocols attempt to mitigate the effects of fluctuating bandwidth and high delay by dynamically adjusting the video quality (bitrate) based on observed network conditions. However, these protocols may not be perfectly efficient in reacting to unpredictable changes in network conditions (Yin et al., 2015). Network congestion further exacerbates latency, as increased traffic delays and packet loss.

2.2.2 Server Performance and Content Placement

Server performance and content placement are crucial to minimizing latency.

1. **Server Load:** High server load leads to increased processing times, delayed segment delivery, and increased latency (Akhshabi et al., 2013). Efficient server configurations and load balancing techniques are essential for distributing the load across multiple servers and minimizing the impact of high server load.

2. **Content Placement:** Content placement strategies (i.e., where to store video content) play a critical role in minimizing latency (Golrezaei et al., 2013). Content delivery networks (CDNs) improve latency by strategically locating content servers closer to users, reducing the distance data needs to travel to reach the client.
3. **Server Processing:** The time taken for the server to process incoming requests and prepare segments for delivery also contributes to latency. Efficient server-side processing techniques, including optimized video encoding and segment generation, are critical for minimizing this component of latency. Server-side traffic shaping and load balancing can also play a crucial role in minimizing latency (Akhshabi et al., 2013).

2.2.3 Video Encoding and Segmentation

Video encoding and segmentation methods significantly influence latency.

1. **Video Encoding Schemes:** The complexity of video encoding schemes directly impacts processing time, affecting latency. More complex encodings (e.g., higher resolution, higher bitrates) demand increased processing power, resulting in increased latency, especially on low-power mobile devices (Yin et al., 2015). (Note: This study assumes constant encoding parameters for ABR evaluation to isolate network effects).
2. **Segment Length:** The length of video segments affects both the frequency of quality adaptation and overhead. Shorter segments enable more frequent adjustments to video quality based on network conditions, improving the viewing experience. However, they also introduce overhead due to increased signaling and increased transmission of smaller packets (Dobrian et al., 2011). Longer segments reduce overhead but result in slower adaptation to changes in network conditions. The optimal segment length involves a complex trade-off between these two factors. (Note: This study uses a fixed segment duration, as specified in Chapter 3, to provide a consistent basis for ABR evaluation).

2.2.4 Client-Side Factors

Client-side factors play a critical role in managing latency.

1. **Buffering:** Clients use buffers to store incoming video segments, smoothing out short-term fluctuations in network bandwidth. Insufficient buffer size leads to buffering delays and stutters, which negatively impact the viewing experience (Miller et al., 2012).
2. **Rate Adaptation:** Adaptive bitrate (ABR) algorithms dynamically adjust the video quality (bitrate) based on observed network conditions and client device capabilities (Yin et al., 2015). Efficient ABR algorithms are crucial for maintaining smooth playback while minimizing latency, but they also add complexity. The design and evaluation of such ABR algorithms, particularly those sensitive to RTT, are a central focus of this research.
3. **Device Processing Power:** The processing power of the client device significantly impacts the speed of video decoding and processing, which directly influences playback latency. Low processing power on mobile devices exacerbates latency issues (Yin et al., 2015). Client-side caching can help reduce latency by storing frequently accessed video

segments locally, but its effectiveness is limited by device storage capacity. (Note: This study assumes idealized and consistent client device processing power to isolate the impact of network conditions and ABR logic).

Client-side buffer management and playback algorithms are also essential for efficient latency management. These must be optimized to ensure smooth playback and mitigate the impact of factors such as network jitter and varying bandwidth conditions (Miller et al., 2012).

2.3 Existing Optimization Techniques for Video Streaming Latency

2.3.1 Network-Level Optimizations

Network-level optimizations aim to improve the underlying network infrastructure to reduce latency. Key techniques include:

1. **Multipath Streaming:** This technique utilizes multiple network paths to transmit video data, improving bandwidth utilization and reducing latency. By diversifying the transmission paths, multipath streaming can mitigate the impact of congestion or failures on a single path, ensuring smoother playback (Qian et al., 2016). However, it adds complexity in terms of path selection, management, and coordination, and may not always yield significant latency improvements.

2. **Bandwidth Aggregation:** This approach combines multiple network interfaces (e.g., Wi-Fi and cellular) to increase the overall available bandwidth for video streaming. This improves the robustness of the connection and enhances the download speed, leading to reduced latency. However, it requires managing multiple interfaces and protocols, potentially impacting performance and increasing complexity. It may also suffer from interference issues with specific links or configurations (Golrezaei et al., 2013).

3. **Proactive Caching and Pre-fetching:** This technique reduces latency by storing popular content closer to users, minimizing data retrieval time. Proactive caching involves predicting user demand and pre-fetching content to edge servers or local caches. While this can significantly improve latency for popular content, predicting user demand accurately is challenging and maintaining cache consistency across a distributed network may pose difficulties. Inefficient cache management strategies could lead to poor performance and wasted resources. The implementation complexity for dynamic caching is high (Golrezaei et al., 2013).

2.3.2 Server-Side Optimizations

Server-side optimizations focus on enhancing server performance and content delivery efficiency to minimize latency. Key techniques include:

1. **Optimized Content Placement:** Strategically placing content across multiple servers reduces the average distance data needs to travel to reach the user, minimizing latency. CDNs employ this approach extensively by distributing content across geographically dispersed data centers (Dahlman et al., 2014). However, efficient content placement requires sophisticated algorithms and detailed knowledge of network topology and user access patterns. Poor placement strategies could result in longer latency for some users.

2. **Load Balancing and Traffic Shaping:** Load balancing distributes streaming requests evenly across multiple servers, preventing any single server from being overloaded and minimizing latency. Traffic shaping techniques control data transmission rates, ensuring that bandwidth is allocated fairly and preventing congestion (Akhshabi et al., 2013). Sophisticated load balancing algorithms and traffic shaping techniques are needed to manage fluctuating network demands and maximize efficiency.

3. **Edge Computing and CDNs:** Edge computing pushes processing and content closer to end users, reducing latency significantly. CDNs combine edge computing with distributed server infrastructure to improve content delivery efficiency and reduce latency. The benefits of edge computing are limited by the need to deploy and maintain extensive edge server infrastructure. Efficient management and coordination of the edge servers become extremely critical (Akhshabi et al., 2013).

(Note: Similar to network-level, these are contextual. This study's ABR operates at the client, reacting to the network conditions shaped by such server-side strategies.)

2.3.3 Client-Side Optimizations

Client-side optimizations improve the client's ability to manage video playback and reduce the impact of network fluctuations to minimize latency. Key techniques include:

1. **Improved Adaptation Strategies:** Adaptation strategies dynamically adjust the video quality (bitrate) based on observed network conditions and client capabilities. Algorithms react swiftly to network changes, prioritizing QoE metrics such as startup delay and buffering frequency. The complexity of adaptation algorithms increases with the goal of improving efficiency and user experience and accurate prediction of future network conditions is critical (Yin et al., 2015). Many ABR algorithms primarily use bandwidth estimations (e.g., BOLA by Spiteri et al., 2016) or buffer occupancy (e.g., buffer-based approaches like that in Akhshabi et al., 2011) to make decisions.
2. **Efficient Buffer Management:** Optimizing buffer sizes, pre-fetching segments, and making dynamic buffer adjustments are essential for minimizing buffering delays and maintaining continuous playback (Miller et al., 2012). Efficient buffer management

involves a complex tradeoff between minimizing latency and managing memory usage on the client device.

3. **Low-Latency ABR Algorithms:** Low-latency ABR algorithms prioritize minimizing playback latency. These may involve heuristics that quickly assess network conditions or advanced techniques like machine learning to predict future network conditions and optimize bitrate decisions in real-time (Dahlman et al., 2014). Some approaches explicitly incorporate delay measurements or RTT estimates (e.g., an area this research explores with `LatencyAwareAbr`), while others focus on aggressive buffer targets or faster quality down-switching. Developing and implementing low-latency ABR algorithms can be challenging given the complex interactions between network conditions and user experience.

2.3.4 Emerging Technologies and Techniques

Emerging technologies offer further potential for minimizing latency in video streaming:

1. **Machine Learning for Latency Prediction:** Machine learning models can analyze historical network data to predict future latency and/or bandwidth and optimize bitrate decisions. These models can improve the accuracy and timeliness of network condition assessment, enabling more proactive and efficient adaptation to prevent service disruptions. Developing accurate predictive models is computationally intensive and highly dependent on the quality and volume of data (Balachandran et al., 2013; Mao et al., 2017). While the ABR evaluated in this study (`LatencyAwareAbr`) uses simpler heuristic predictors, the potential for ML integration represents a significant avenue for future enhancement.
2. **Viewport Prediction and Tile-Based Streaming:** For 360-degree and VR videos, this technique prioritizes tiles within the predicted viewing area to reduce the amount of data that needs to be transmitted, minimizing latency. This approach requires efficient viewport prediction algorithms which add complexity (Akhshabi et al., 2013). Accurate viewport prediction is crucial for minimizing latency, but is highly dependent on the accuracy of user head movement prediction. (Note: While highly relevant for overall latency in VR, specific tile-based optimizations are outside the scope of the ABR logic evaluated in this study, which focuses on general ABR latency.)

2.4 Gaps in the Literature on Video Streaming Latency Optimization

2.4.1 Latency-Aware Adaptation Needs

Current adaptive streaming protocols (DASH, HLS) often prioritize high bitrates to maximize video quality, potentially overlooking the importance of minimizing latency. In mobile network environments with fluctuating bandwidth and high delay, this prioritization of bitrate over latency can lead to suboptimal Quality of Experience (QoE), manifesting as buffering, startup delays, and frequent quality changes. Existing solutions prioritize high bitrates over low latency, leading to suboptimal QoE in mobile networks (Petrangeli et al., 2017). There is a clear need for algorithms that explicitly prioritize low latency by robustly incorporating factors like RTT into their core decision-making process, while maintaining acceptable video quality, particularly for delay-sensitive applications like video conferencing and interactive gaming. These algorithms should specifically address the challenges of unpredictable network conditions inherent in mobile environments.

2.4.2 Network Delay versus Bandwidth Focus

Much of the existing research on video streaming latency focuses primarily on bandwidth optimization, overlooking the significant impact of network delay (particularly RTT). While bandwidth limitations can impact video quality and cause buffering, network delay directly contributes to latency, regardless of available bandwidth. Network delay (RTT) is often the dominant factor affecting video streaming performance in mobile environments where delays exceeding critical thresholds significantly limit achievable bitrates and video quality (Costa Filho et al., 2016). In contrast, bandwidth in mobile networks, particularly broadband networks, is highly variable, but this variability is often less impactful than sustained high delay on user experience. More research is needed to understand the complex interplay between bandwidth and delay (RTT) in different network environments (Oyman & Singh, 2012), and to develop optimization strategies that effectively manage both aspects to minimize latency. This study aims to contribute to this area by specifically isolating and evaluating the impact of RTT.

2.4.3 Device and Network Heterogeneity

Most existing video streaming optimization algorithms assume homogenous network conditions and device capabilities, ignoring the significant diversity in real-world scenarios (Petrangeli et al., 2017). Mobile devices vary widely in processing capabilities, screen sizes, and resolutions, and mobile networks exhibit significant variability in bandwidth and latency (including RTT). The lack of consideration for these heterogeneities means that one-size-fits-all algorithms often fail to provide optimal QoE. This study specifically investigates the impact of network RTT heterogeneity on client performance and playback discrepancy. Algorithms and architectures should be designed to explicitly take into account this device and network heterogeneity, tailoring optimization strategies to the individual capabilities and constraints of each device and network condition. A more holistic approach to optimization is required to consider how different devices may demand different quality levels at different network conditions.

2.4.4 Emerging Needs in 360-Degree and VR Streaming

The rapidly growing popularity of 360-degree and VR video streaming introduces unique challenges for latency optimization. These immersive experiences require significantly higher bandwidths and more sophisticated delivery techniques to achieve acceptable QoE. Viewport prediction and tile-based streaming, which aims to deliver only the visible parts of the video, are promising approaches for optimizing bandwidth usage. However, viewport prediction algorithms must be very accurate to avoid significant quality degradation (Akhshabi et al., 2013). Further, adaptation strategies must be tailored to handle the specific characteristics of VR videos, addressing issues such as spatial segmentation and quality zone selection, which necessitates more research to develop efficient and robust algorithms (Fan et al., 2017). Adaptive algorithms that prioritize low latency for VR and 360-degree video, that efficiently respond to changing network conditions while minimizing buffering and quality changes are needed. Understanding the fundamental impact of RTT on general ABR performance, as explored in this study, is a foundational step towards addressing these more complex scenarios.

References

i. Akhshabi, S., Anantakrishnan, L., Dovrolis, C., & Begen, A. C. (2013). Server-based traffic shaping for stabilizing oscillating adaptive streaming players. *Proceedings of the 23rd ACM Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV '13)*, 19–24.

ii. **Akhshabi, S., Begen, A. C., & Dovrolis, C. (2011). An experimental evaluation of rate-adaptation algorithms in adaptive streaming over HTTP. Proceedings of the 2nd Annual ACM Multimedia Systems Conference (MMSys '11), 157–168.* (Added here as it's relevant for buffer-based approaches)*

iii. Balachandran, A., Sekar, V., Akella, A., Seshan, S., Stoica, I., & Zhang, H. (2013). Developing a predictive model of quality of experience for Internet video. *Proceedings of the ACM SIGCOMM Conference*, 339–350.

iv. Dahlman, E., Mildh, G., Parkvall, S., Peisa, J., Sachs, J., Selén, Y., & Sköld, J. (2014). 5G wireless access: Requirements and realization. *IEEE Communications Magazine*, *52*(12), 42–47.

v. da Costa Filho, R. I. T., Lautenschlager, W., Kagami, N., Roesler, V., & Gaspary, L. P. (2016). Network fortune cookie: Using network measurements to predict video streaming performance and QoE. *2016 IEEE Global Communications Conference (GLOBECOM)*, 1–6.

vi. Dobrian, F., Sekar, V., Awan, A., Stoica, I., Chuang, D., Ganjam, A., ... Zhang, H. (2011). Understanding the impact of video quality on user engagement. *Proceedings of the ACM SIGCOMM Conference (SIGCOMM '11)*, 362–373.

vii. Fan, C.-L., Lee, J., Lo, W.-C., Huang, C.-Y., Chen, K.-T., & Hsu, C.-H. (2017). Fixation prediction for 360 video streaming in head-mounted virtual reality. *Proceedings of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV '17)*, 67–72.

viii. Golrezaei, N., Molisch, A. F., Dimakis, A. G., & Caire, G. (2013). Femto-caching and device-to-device collaboration: A new architecture for wireless video distribution. *IEEE Communications Magazine, 51*(4), 142-149.

ix. Li, Z., Zhu, X., Gahm, J., Pan, R., Hu, Y., Begen, A. C., & Oran, D. (2014). Probe and adapt: Rate adaptation for HTTP video streaming at scale. *IEEE Journal on Selected Areas in Communications, 32*(4), 719-733.

x. Mao, H., Netravali, R., & Alizadeh, M. (2017). Neural adaptive video streaming with pensieve. *Proceedings of the Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '17)*, 197-210.

xi. Miller, K., Ramakrishnan, K. K., & Polyzos, G. C. (2012). Buffer-aware adaptation for HTTP adaptive streaming. *2012 IEEE International Conference on Multimedia and Expo Workshops*, 380-385.

xii. Oyman, O., & Singh, S. (2012). Quality of experience for HTTP adaptive streaming services. *IEEE Communications Magazine*, *50*(4), 20–27.

xiii. Petrangeli, S., Swaminathan, V., Hosseini, M., & De Turck, F. (2017). An HTTP/2-based adaptive streaming framework for 360 virtual reality videos. *Proceedings of the 2017 ACM on Multimedia Conference (MM '17)*, 306–314.

xiv. Petrangeli, S., Famaey, J., Claeys, M., Latré, S., & De Turck, F. (2015). QoE-driven rate adaptation heuristic for fair adaptive video streaming. *ACM Transactions on Multimedia Computing, Communications, and Applications*, *12*(2), Article 28.

xv. Qian, F., Ji, L., Han, B., & Gopalakrishnan, V. (2016). Optimizing 360 video delivery over cellular networks. *Proceedings of the 5th workshop on All things cellular: Operations, applications and challenges*, 1-6.

xvi. Sodagar, I. (2011). The MPEG-DASH standard for multimedia streaming over the Internet. *IEEE Multimedia*, *18*(4), 62–67.

xvii. Spiteri, K., Sitaraman, R., & Sparacio, D. (2016). BOLA: Near-optimal bitrate adaptation for online videos. *IEEE/ACM Transactions on Networking, 28*(4), 1-14. *(Example of a notable ABR algorithm, BOLA, which focuses on buffer and is a good contrast to RTT-aware approaches).*

xviii. Yin, X., Jindal, A., Sekar, V., & Sinopoli, B. (2015). A control-theoretic approach for dynamic adaptive video streaming over HTTP. *SIGCOMM Computer Communication Review*, *45*(4), 325–338.